



Centro Universitário Farias Brito
Bacharelado em Ciência da Computação
TCC

CryptoEdu - Simulador de Algoritmo de
Criptografia de Cifra de Blocos com Finalidade
Educacional

Tanielian Viana Barreira

Me. Sérgio Araújo Yunes

Tanielian Viana Barreira

CryptoEdu - Simulador de Algoritmo de Criptografia de Cifra de Blocos com Finalidade Educacional

Trabalho de Conclusão de Curso para o curso
de Ciência da Computação do Centro Univer-
sitário Farias Brito.

Centro Universitário Farias Brito
Bacharelado em Ciência da Computação
TCC

Orientador(a): Me. Sérgio Araújo Yunes

Fortaleza-CE, Brasil

2020

CryptoEdu - Simulador de Algoritmo de Criptografia de Cifra de Blocos com Finalidade Educacional

TANIELIAN VIANA BARREIRA

Esta monografia foi submetida ao curso de Ciência da Computação do Centro Universitário Farias Brito como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Na avaliação da banca, este trabalho obteve conceito 9 (nove), conferido pelos avaliadores da banca e outorgada pelo referido Centro Universitário.

A citação de qualquer trecho desta monografia é permitida, desde que seja feita de acordo com as normas científicas.

Autor(a) Tanielian Viana Barreira

Banca Examinadora:

Orientador(a) Prof(a). Me. Sérgio Araújo Yunes

Examinador(a) Prof(a). Me. José Helano Matos Nogueira

Examinador(a) Prof(a). Me. Maikol Magalhães Rodrigues

Monografia avaliada em 12 de dezembro de 2020

*Dedico este trabalho a todos os que me ajudaram ao longo desta caminhada.
Especialmente aos meus pais, amigos e esposa.*

Agradecimentos

Agradeço ao professor Dr. Jorge Albuquerque por ter me apresentado à Computação, esta área que tanto amo. Agradeço também aos professores Dr. Fani Carvalho, Cleiton Albuquerque, Ricardo Wagner, Maikol Rodrigues. E agradeço especialmente ao meu orientador Me. Sérgio Araújo Yunes por ter me auxiliado na criação deste trabalho, mas principalmente, pela paciência.

Resumo

A manipulação de informações sensíveis, senhas, dinheiro e informações privadas, por meio da *internet* faz com que a segurança da informação seja um ponto que, cada vez, mais deve ser levado em consideração ao se manipular dados, seja ao persistir ou ao trafegar estes. A criptografia, um dos pilares da segurança da informação, por sua vez, é complexa, dificultando assim o seu ensino e, conseqüentemente, sua utilização de maneira adequada ou otimizada. Vale a pena salientar também, que o uso de Objetos de Aprendizagem (OAs) para melhorar o processo de ensino-aprendizagem vem crescendo cada vez mais e as suas áreas de aplicabilidade também. Isso é decorrente dos vários benefícios provenientes da utilização de OAs no ambiente de ensino, em especial no Ensino a Distância (EaD). Baseado no contexto exposto, este trabalho apresenta o resultado de uma pesquisa na área da Informática na Educação, que objetivou criar um Objeto de Aprendizagem simulador capaz de auxiliar no processo de ensino-aprendizagem de algoritmos criptográficos por cifra de bloco. No simulador, são explicadas todas as etapas do algoritmo simulado que, por ser o mais adequado ao ensino, é o S-DES. Além disso, foi realizada uma pesquisa com intuito de avaliar a efetividade do OA no processo de ensino-aprendizagem e esta atestou que o simulador foi capaz de aumentar o conhecimento de alunos e professores sobre a temática abordada.

Palavras-chave: Informática na Educação, Objeto de Aprendizagem (OA), Simulador, Criptografia, S-DES

Abstract

The manipulation of sensitive information, passwords, money, and private information, through the internet, makes information security a point that more and more, must be taken into account when manipulating data, either when persisting or when trafficking these. Cryptography, one of the pillars of information security, in turn, is complex, making it difficult to teach and, consequently, to use in an appropriate or optimized way. It is also worth noting that the use of Learning Objects (LOs) to improve the teaching-learning process has been growing more and more and its areas of applicability as well. This is due to the various benefits arising from the use of LOs in the teaching environment, especially in Distance Learning (DL). Based on the given context, this work presents the result of a research in the field of Computers in Education, which aimed to create a simulator capable of assisting in the teaching-learning process of block cipher cryptographic algorithms. In the simulator, all stages of the simulated algorithm S-DES are explained, the reason for that being the fact that this is the most suitable algorithm for teaching. In addition to that, a survey was carried out in order to assess the effectiveness of the LO in the teaching-learning process and this asserted that the simulator was able to increase the knowledge of students and teachers on the addressed topic.

Keywords: Computers in Education, Learning Object, Simulator, Cryptography, S-DES

"Antes tarde do que nunca"

Lista de Siglas e Abreviações

3DES *Triple Data Encryption Standard*. 32–34

AES *Advanced Encryption Standard*. 32

AVAs Ambientes Virtuais de Aprendizagem. 17

CAI *Computer Aided Instruction*. 18

DES *Data Encryption Standard*. 13, 32–34

EaD Ensino a Distância. 5, 17, 69

EFF *Electronic Frontier Foundation*. 32

FBuni Centro Universitário Farias Brito. 64

GCHQ *Government Communications Headquarters*. 19

NBS *National Bureau of Standards*. 32

NSA *National Security Agency*. 32

OA Objeto de Aprendizagem. 5, 13, 17, 69

OAs Objetos de Aprendizagem. 5, 13, 14, 69

PEC Programas Educacionais por computador. 18

S-DES *Simplified Data Encryption Standard*. 9, 34–38, 40

TCC Trabalho de Conclusão de Curso. 57

UNIFOR Universidade de Fortaleza. 64

Lista de ilustrações

Figura 1 – Simulador: CyberChef	19
Figura 2 – Simulador: CrypTool 2	20
Figura 3 – Simulador: JCrypTool	21
Figura 4 – Simulador: S-DES Simulator	22
Figura 5 – Simulador: S-DES Simulator’s warning	22
Figura 6 – Simulador: S-DES Simulator	23
Figura 7 – Simulador: S-DES Simulator	24
Figura 8 – Criptografia Simétrica	26
Figura 9 – Exemplo de <i>Rail fence</i>	26
Figura 10 – Exemplo de cifra de César	27
Figura 11 – Imagem da máquina Enigma	27
Figura 12 – Fluxo da mensagem criptografada quando o foco é confidencialidade . .	28
Figura 13 – Fluxo da mensagem criptografada quando o foco é autenticidade . . .	29
Figura 14 – Fluxo da mensagem criptografada quando o foco é confidencialidade e autenticidade	30
Figura 15 – Cifra de bloco ideal	31
Figura 16 – Estrutura do S-DES	35
Figura 17 – Geração das Chaves do S-DES	36
Figura 18 – Detalhamento da criptografia do S-DES	37
Figura 19 – Visualização em um <i>browser</i> de dispositivo móvel	39
Figura 20 – <i>UI Breakpoints</i>	40
Figura 21 – Estrutura da interface	41
Figura 22 – Cabeçalho	41
Figura 23 – Conteúdo	42
Figura 24 – Navegação no último passo	42
Figura 25 – Rodapé	42
Figura 26 – Executor bit a bit no início da execução da etapa	43
Figura 27 – Executor bit a bit durante a execução da etapa	44
Figura 28 – Executor bit a bit no fim da execução da etapa	44
Figura 29 – Explicação expandida	45
Figura 30 – Etapa P10 - Permutação de 10 bits	46
Figura 31 – Etapa LS-1 - <i>Circular Left Shift</i> de 1 posição	47
Figura 32 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-1	47
Figura 33 – Chave K_1 - Geração da primeira chave K_1	48
Figura 34 – Etapa LS-2 - <i>Circular Left Shift</i> de 2 posições	48
Figura 35 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-2	49

Figura 36 – Chave K_2 - Geração da segunda chave K_2	49
Figura 37 – Passo IP (<i>Initial Permutation</i>) - Permutação Inicial	50
Figura 38 – L (esquerda) & R (direita)	50
Figura 39 – Passo f_K - Função que usa uma chave	51
Figura 40 – Etapa E/P - Permutação de Expansão	51
Figura 41 – Etapa XOR - OU exclusivo com K_1	52
Figura 42 – Etapa S0 & S1 - Substituições S0 e S1	53
Figura 43 – Etapa P4 - Permutação de 4 bits	54
Figura 44 – Etapa XOR - OU exclusivo com L	54
Figura 45 – Resultado de f_K	55
Figura 46 – Passo SW - Troca	55
Figura 47 – Passo IP^{-1} - Permutação Inicial Inversa	56
Figura 48 – Resultado	57
Figura 49 – <i>Feedback request</i>	57
Figura 50 – Início do questionário	59
Figura 51 – Enquadramento do usuário	60
Figura 52 – Conhecimento antes da utilização do simulador	61
Figura 53 – Conhecimento depois da utilização do simulador	61
Figura 54 – Efetividade do aprendizado	62
Figura 55 – Melhorias nas descrições das etapas	63
Figura 56 – Melhorias nas execuções das etapas	64
Figura 57 – Enquadramento do usuário	65
Figura 58 – Conhecimento antes vs. depois da utilização do simulador por enqua- dramento	66
Figura 59 – Efetividade do aprendizado	67
Figura 60 – Melhorias no simulador	68

Sumário

1	INTRODUÇÃO	13
2	FUNDAMENTAÇÃO TEÓRICA	15
2.1	Informática na educação	15
2.2	Objetos de aprendizagem	16
2.3	Ferramentas de apoio ao ensino	17
2.4	Simuladores voltados ao ensino	18
2.5	Simuladores de criptografia	18
2.5.1	CyberChef	19
2.5.2	CrypTool	20
2.5.2.1	CrypTool 2	20
2.5.2.2	JCrypTool	21
2.5.3	S-DES Simulator (<i>Android App</i>)	22
2.5.4	S-DES Simulator (<i>online</i>)	23
2.5.5	S-DES Simulator (Win)	24
2.6	Criptografia	24
2.6.1	Criptografia Simétrica	25
2.6.2	Criptografia Assimétrica	28
2.7	Cifra de blocos	30
2.8	DES	31
2.9	3DES	33
2.10	S-DES	34
3	O SIMULADOR	38
3.1	Informações técnicas	38
3.2	Estrutura da interface	40
3.2.1	Cabeçalho	41
3.2.2	Conteúdo	42
3.2.3	Rodapé	42
3.2.4	Executor bit a bit	43
3.3	Interface do conteúdo	44
3.3.1	Tela inicial - entrada de dados	45
3.3.2	Passo Chaves - Geração das chaves	46
3.3.2.1	Etapa P10 - Permutação de 10 bits	46
3.3.2.2	Etapa LS-1 - <i>Circular Left Shift</i> de 1 posição	47
3.3.2.3	Etapa P8 - Permutação de 8 bits sobre o resultado de LS-1	47

3.3.2.4	Resultado Chave K_1	48
3.3.2.5	Etapa LS-2 - <i>Circular Left Shift</i> de 2 posições	48
3.3.2.6	Etapa P8 - Permutação de 8 bits sobre o resultado de LS-2	49
3.3.2.7	Resultado Chave K_2	49
3.3.3	Passo IP - Permutação Inicial	50
3.3.4	Passo f_K - Função que usa uma chave	51
3.3.4.1	Etapa E/P - Permutação de Expansão	51
3.3.4.2	Etapa XOR - OU exclusivo com K_1	52
3.3.4.3	Etapa S0 & S1 - Substituições S0 e S1	53
3.3.4.4	Etapa P4 - Permutação de 4 bits	54
3.3.4.5	Etapa XOR - OU exclusivo com L	54
3.3.4.6	Resultado de f_K	55
3.3.5	Passo SW - Troca	55
3.3.6	Passo f_K - Função que usa a outra chave	56
3.3.7	Passo IP^{-1} - Permutação Inicial Inversa	56
3.3.8	Tela final - Resultado	57
3.4	Comparação com outros simuladores criptográficos	57
4	PESQUISA DE REAÇÃO	59
4.1	Perguntas	59
4.1.1	Enquadramento do usuário	60
4.1.2	Conhecimento antes e depois da utilização do simulador	60
4.1.3	Efetividade do aprendizado	62
4.1.4	Melhorias no simulador	62
4.2	Resultados da pesquisa	64
4.2.1	Enquadramento do usuário	65
4.2.2	Conhecimento antes e depois da utilização do simulador	66
4.2.3	Efetividade do aprendizado	67
4.2.4	Melhorias no simulador	68
5	CONCLUSÕES E TRABALHOS FUTUROS	69
	REFERÊNCIAS	70

1 Introdução

Várias aplicações da Ciência da Computação possuem como necessidade o conhecimento de como funcionam os métodos e técnicas da Criptografia. Essa necessidade decorre do fato de que a *Internet* (principalmente utilizada em dispositivos móveis), definitivamente, está inserida na vida cotidiana das pessoas e novos costumes estão sendo adotados pela sociedade. O comércio eletrônico, as operações bancárias de transferência eletrônica de fundos, o uso do cartão de crédito como moeda de plástico, por exemplo, se constituem na realidade do convívio social, independente de qual seja a sua posição social. Tornando a segurança da informação um ponto cada vez mais importante ao se manipular/trafegar dados.(SILVA, 2009) (SILVA; JUNIOR, 2012)

A alta complexidade dos algoritmos usados nas aplicações da Criptografia faz com que o processo de ensino-aprendizagem destes nem sempre seja eficiente, principalmente se os algoritmos forem ensinados e analisados na sua complexidade total (SILVA, 2009) (SILVA; JUNIOR, 2012). A academia apresenta aos profissionais em formação na área de Ciência da Computação representações simplificadas do mundo real, embora ainda próximas do mundo real, com o intuito de facilitar o manuseio dessa representação no ambiente de estudo (MAIA, 2001) (MAIA; PACHECO, 2003) (KIOKI; SANTIAGO; SOARES, 2008). Naturalmente, a abordagem de cunho pedagógico precisa preservar os aspectos fundamentais da arquitetura de cada algoritmo (GARMPIS, 2011) (LOPES, 2012).

Outro aspecto importante é a crescente utilização de Objetos de Aprendizagem (OAs) em diversas áreas do meio acadêmico devido aos vários atributos pedagógicos existentes neles, possibilitando o estímulo de diferentes procedimentos cognitivos nos alunos como a observação, a comparação, a análise, a elaboração de hipóteses, a memorização, a checagem ou a manipulação de dados. Dentre suas principais características se destacam a possibilidade de auxiliar tanto o ensino presencial quanto o à distância, a possibilidade de o aluno utilizar o OA fora do horário escolar, normalmente são compatíveis com várias plataformas e normalmente são graficamente ricos, permitindo assim uma maior interação entre o usuário e o OA. (BARBOSA; SCORTEGAGNA, 2015)

Analisando as citações acima apresentadas, este trabalho apresenta o resultado de uma pesquisa na área da Informática na Educação que objetivou criar um OA simulador capaz de auxiliar no processo de ensino-aprendizagem de algoritmos criptográficos por cifra de bloco, nas disciplinas que abordem a criptografia. A ferramenta possibilita a simulação da execução da versão simplificada do algoritmo criptográfico de chave simétrica conhecido como *Data Encryption Standard* (DES). Este simulador permite que o algoritmo seja

executado passo a passo, de maneira sequencial ou não, e até mesmo cada etapa dentro de um passo, isoladamente. Cada etapa possui sua explicação e execução bit a bit. Esta é apresentada na seção 3.

Além disso, foi realizada uma pesquisa de campo com intuito de avaliar a efetividade da ferramenta no processo de ensino-aprendizagem com um grupo diversificado de entrevistados, incluindo alunos e professores. Os resultados dessa pesquisa são apresentados na seção 4.

A seção 2 apresenta uma revisão bibliográfica incluindo um levantamento dos OAs e uma visão geral sobre cifra de blocos e seus algoritmos. Por fim, temos as conclusões encontradas neste trabalho e sugestões de trabalhos futuros, na seção 5.

2 Fundamentação Teórica

Inicia-se com uma explicação geral sobre o enquadramento desse trabalho no meio acadêmico com as seções 2.1 Informática na educação e 2.2 Objetos de aprendizagem.

Na seção 2.3 Ferramentas de apoio ao ensino, se dá uma visão geral sobre o que são e qual o ganho que elas trazem para os alunos. A seção, 2.4 Simuladores voltados ao ensino, traz uma visão geral sobre simuladores e uma análise de simuladores voltados ao ensino de uma forma geral e a seção 2.5 descreve os simuladores de criptografia atuais.

A seção 2.6 Criptografia discorre sobre criptografia e suas aplicabilidades, assim como seus dois tipos, 2.6.1 Criptografia Simétrica e 2.6.2 Criptografia Assimétrica. Apresentam-se suas definições, seus cenários de utilização, e trazem alguns exemplos de algoritmos sobre seus respectivos ramos da criptografia. A seção 2.7 Cifra de blocos fala sobre esse método de criptografia simétrica no qual é feita uma visão geral sobre seu funcionamento e benefícios obtidos por sua utilização.

As seções 2.8 DES, 2.9 3DES e 2.10 S-DES tratam sobre estes algoritmos de criptografia simétrica baseados em cifra de blocos, os quais trazem uma visão geral e uma explicação de cada algoritmo dando uma ênfase maior no S-DES por ser o algoritmo utilizado na ferramenta desenvolvida.

2.1 Informática na educação

A educação abrange dar e receber conhecimento. Qualquer sociedade possui esse cenário, que é responsável pela manutenção e propagação às gerações que se seguem, da cultura necessária à convivência de um membro na sua sociedade (HAMAWAKI; PELEGRINI, 2009).

Dentre as formas de aprendizagem, são citadas duas teorias importantes por Ausubel (AUSUBEL; NOVAK; HANESIAN, 1980), a aprendizagem significativa e a mecânica.

A primeira diz respeito à ideia de que a aprendizagem é um processo pelo qual uma informação recém-adquirida relaciona-se com um aspecto importante da estrutura de conhecimento do aluno, ou seja, quando a nova informação adquirida vincula-se em conhecimentos relevantes previamente adquiridos na estrutura cognitiva do aluno. A aprendizagem significativa é onde o aluno realmente aprende.

A aprendizagem mecânica por sua vez ocorre quando o novo conhecimento se associa pouco ou não se associa a algum conhecimento importante já existente na estrutura cognitiva do aprendiz. Quando isso ocorre o conhecimento adquirido recentemente é

gravado de maneira arbitrária. Essa aprendizagem é inviável quando um aluno recebe um novo conhecimento em uma área de conhecimentos nova para ele. Ou seja, aprendizagem mecânica acontece somente até que alguns elementos do conhecimento pré-adquirido, relevantes a novas informações na mesma área de conhecimento, existam na estrutura cognitiva do aluno e possam ser utilizadas como base para a aprendizagem significativa. E é através dessa aprendizagem que se inicia a criação, no aluno, de uma estrutura cognitiva mais complexa, onde ele deixa de aprender de forma passiva e mecânica e passa a assimilar o conteúdo de forma mais clara e não linear (AUSUBEL; NOVAK; HANESIAN, 1980).

Muitas vezes a transmissão do conhecimento é árdua e complexa. Tendo em vista essa dificuldade, almeja-se a utilização de bons recursos didáticos que melhorem o desempenho docente. (SOUZA, 2007). Uma das considerações que devem ser levadas em conta pelo educador como critério de escolha de um recurso didático é que sua utilização deve preencher as lacunas existentes no ensino tradicional e viabilizar ao aluno a ampliação tanto da capacidade de reter conhecimento quanto a sua visão, e ainda estimular o ensino docente. (TRIVELATO; OLIVEIRA, 2006)

Sons, imagens, construção de maquetes e até o uso de materiais lúdicos, esses simples elementos podem ser aplicados como recursos didáticos. Quando o corpo docente se utiliza de tipos variados de recursos didáticos ele não somente faz com que o ensino seja mais interessante do que o ensino tradicional, como também pode propiciar melhores resultados (PARRA, 1985) (SOUZA, 2007) (CASTOLDI; POLINARSKI, 2009). Dentre os recursos didáticos que podem ser utilizados existem vários tipos, como: livros, apostilas, artigos, quadro e giz, apresentações, maquetes, trabalhos acadêmicos, softwares, ilustrações, filmes, músicas, exercícios físicos, excursões, brincadeiras e outros (FERREIRA, 2007). A brincadeira Adedonha, por exemplo, já teve sua eficiência comprovada como ferramenta prático-pedagógica no processo de ensino-aprendizagem (SILVA *et al.*, 2018).

A abordagem do uso do computador e/ou dispositivos eletrônicos similares como recurso pedagógico utilizado pelo docente no processo de ensino do corpo discente é conhecido como Informática na educação. (KENSKI, 2007)

2.2 Objetos de aprendizagem

"...definimos objetos de aprendizagem como sendo recursos digitais dinâmicos, interativos e reutilizáveis em diferentes ambientes de aprendizagem elaborados a partir de uma base tecnológica. Desenvolvidos com fins educacionais, eles cobrem diversas modalidades de ensino: presencial, híbrida ou à distância; diversos campos de atuação: educação formal, corporativa ou informal; e, devem reunir várias características como durabilidade, facilidade para atualização, flexibilidade, interoperabilidade, modularidade, portabilidade, entre outras. Eles ainda apresentam-se como unidades autoconsistentes de pequena extensão e

fácil manipulação, passíveis de combinação (essa combinação dá-se, principalmente, no sentido objeto-mídia, mas o contrário também é válido dependendo da mídia, pois não são todas que oferecem a característica de hiperligação) com outros objetos educacionais ou qualquer outra mídia digital (vídeos, imagens, áudios, textos, gráficos, tabelas, tutoriais, aplicações, mapas, jogos educacionais, animações, infográficos, páginas web) por meio da hiperligação. Além disso, um objeto de aprendizagem pode ter usos variados, seu conteúdo pode ser alterado ou reagregado, e ainda ter sua interface e seu layout modificado para ser adaptado a outros módulos ou cursos. No âmbito técnico, eles são estruturas autocontidas em sua grande maioria, mas também contidas, e marcadas por identificadores denominados metadados." (AUDINO, 2012)

Em suma, um Objeto de Aprendizagem (OA) é um recurso digital utilizado para suportar o ensino. Imagens, vídeos, softwares e animações podem ser considerados OAs contanto que sejam voltados ao ensino. Já os Ambientes Virtuais de Aprendizagem (AVAs) por sua vez não são considerados OAs pois esses, apesar de gerenciar e até armazenar OAs, não são utilizados diretamente na aprendizagem. (WILEY, 2002) (BRAGA, 2014)

Apesar do AVA ser o principal instrumento mediador em um sistema de Ensino a Distância (EaD) isso não exclui sua utilização como suporte para o ensino presencial (BELMONTE; GROSSI, 2010).

2.3 Ferramentas de apoio ao ensino

Ainda é um desafio a criação de OAs pois geralmente o conhecimento pedagógico é detido por um profissional da área de ensino e este normalmente não possui o conhecimento técnico avançando necessário para o desenvolvimento de OAs (BRAGA, 2015).

Há também OAs que podem ser customizadas e aplicadas para várias áreas de ensino, como é o caso do jogo *Minecraft: Education Edition* que já foi aplicado no ensino de Biologia, Ecologia, Física, Química, Geologia, Geografia (SHORT, 2012) e até Cibersegurança (GEARY; RONKE; GEARY, 2019), .

Existe um problema comum entre professores e alunos de disciplinas do curso de Ciência da Computação: Existe uma dificuldade em demonstrar a real dinâmica dos eventos computacionais. Mesmo que o mestre tenha conhecimento e didática, que o aluno dedique raciocínio e atenção, nem sempre é possível se obter a total compreensão dos conceitos ensinados diante da abordagem destes nas disciplinas.

O computador não deve ser o substituto do professor, mas sim, tomar o papel de ferramenta educacional auxiliando o aprendizado. Com isso o papel do professor passa a ser mais produtivo. Primeiramente, ferramentas voltadas ao ensino, já se utilizando do computador, se baseavam na máquina de *B. F. Skinner*, onde se utilizava a ideia

de instrução programada, que consistia de dividir o conteúdo total em pequenas partes lógicas e sequenciais. Somente durante a década de 1960, foi dado início ao conceito de *Computer Aided Instruction* (CAI), ou seja, Programas Educacionais por computador (PEC). A real dispersão do CAI só aconteceu com popularização dos microcomputadores, possibilitando assim, a criação de diversos tipos de ferramentas de ensino, como, tutoriais, exercício-e-prática, avaliações, jogos, e simulações (HAMAWAKI; PELEGRINI, 2009).

2.4 Simuladores voltados ao ensino

Algumas situações são tão singulares, que se preparar para tais torna-se difícil. E portanto, com o intuito de proporcionar a alguém um conhecimento maior sobre estas situações os simuladores foram criados.

A simulação é uma representação do mundo real. As primeiras simulações foram criadas com o intuito de disponibilizar um ambiente seguro para situações que envolvessem risco ao humano. Dentre elas podemos citar simulações de viagens e mergulhos profundos. Posteriormente, foram criadas com o intuito de se obter uma economia, seja ela de tempo e/ou dinheiro, que é o caso da indústria automobilística e aviação.

Diversas são as áreas de conhecimento que fazem uso das simulações (BANKS; II; NELSON, 2009). Na ciência da computação há uma disponibilidade de simuladores, cada um atendendo a uma respectiva disciplina, assim como redes de computadores, arquitetura de computadores, técnicas de programação e sistemas operacionais.

Os simuladores possuem um potencial bem maior do que as outras ferramentas de ensino citadas acima. A aplicação de simulações no universo acadêmico permite que o aluno desenvolva hipóteses, teste-as, analise os resultados e concretize seus conhecimentos. Simuladores devem ser utilizados como ferramenta complementar as aulas lecionadas pelo professor.

Mesmo com todas as vantagens envolvidas, o desenvolvimento de um simulador não é algo trivial. Uma vez que a ferramenta possui um cunho pedagógico, a utilização de recursos multimídia para tornar a simulação mais fiel à realidade é muito importante, e a utilização destes recursos não é algo de simples desenvolvimento.

Quando o aluno possui pouco conhecimento na área, os simuladores são indicados como melhor ferramenta de auxílio. Principalmente, se forem para auxiliar cursos de extensão e graduação (MAIA, 2001) (MAIA; PACHECO, 2003).

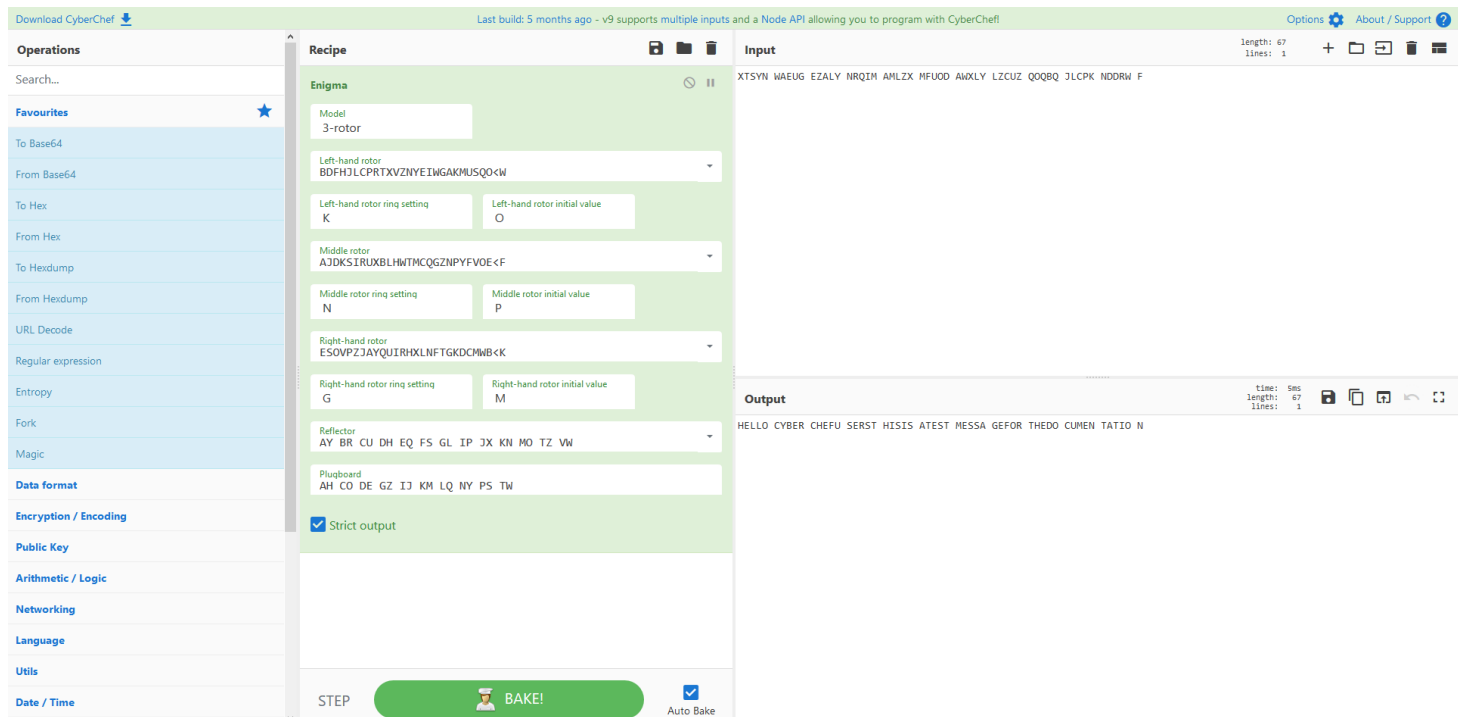
2.5 Simuladores de criptografia

Essa seção descreve alguns simuladores de criptografia.

Além dos simuladores descritos nas seções seguintes foram encontrados o "Cryptotutor: *An educational tool for learning modern cryptography*" (LUBURIĆ *et al.*, 2016) e o CET - *Cryptographic Education Tool* (ABUZAIID *et al.*, 2011). Porém um estudo sobre estes não pôde ser feito pois os simuladores não estão disponíveis para uso e suas publicações científicas não estão disponíveis publicamente.

2.5.1 CyberChef

Figura 1 – Simulador: CyberChef



Fonte: (GCHQ, 2020)

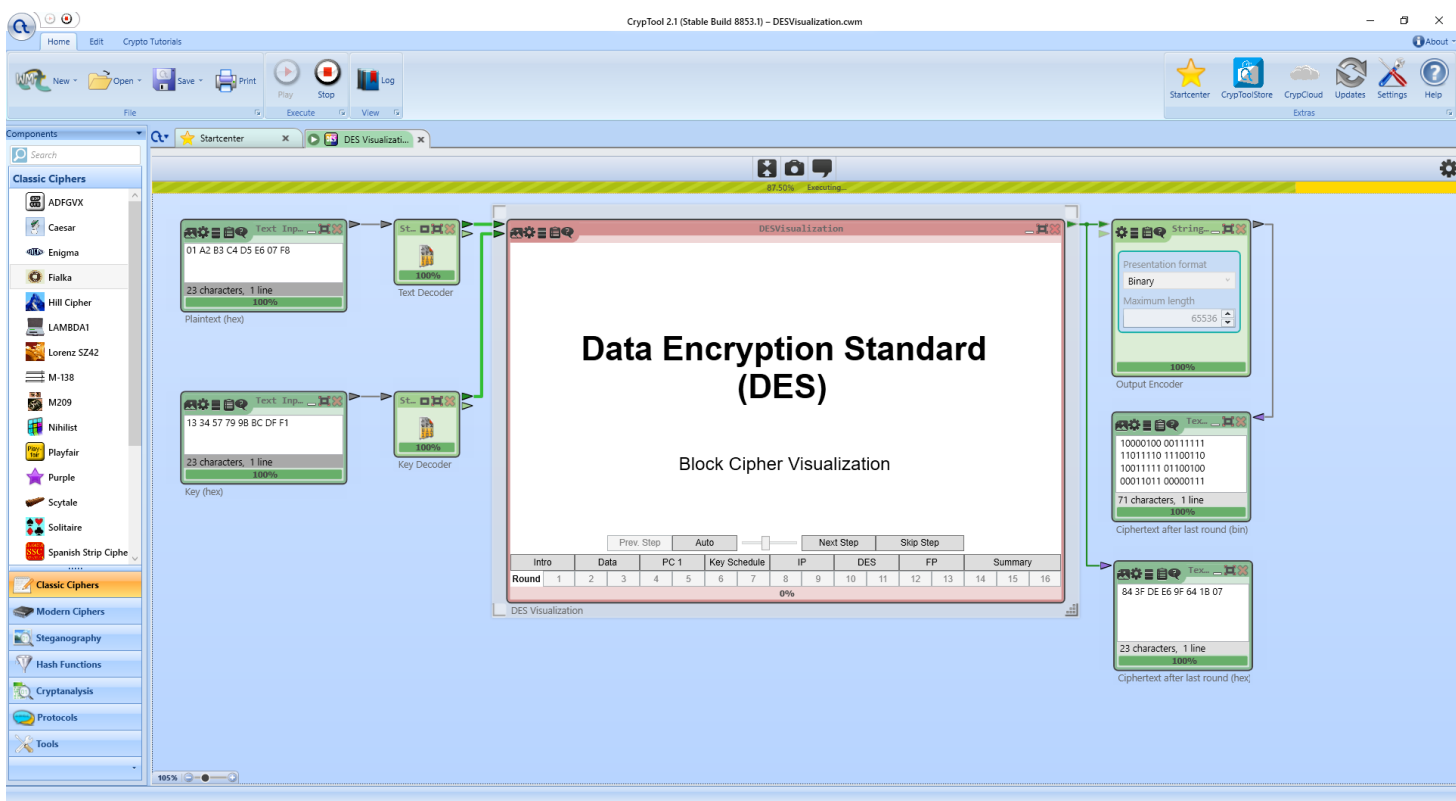
O CyberChef (Figura 1) trata-se de um simulador da máquina enigma criado pela *Government Communications Headquarters* (GCHQ), o centro de inteligência britânica. Ele está disponibilizado no idioma inglês e permite a utilização de vários algoritmos simples (como *ToBase64String* e etc.), podendo inclusive encadear a execução destes sobre o *input*. Nele pode-se informar os parâmetros de entrada que ele utiliza para mostrar o resultado da execução do algoritmo. Ele não é voltado ao ensino e sim para testes dos algoritmos com entradas customizadas. Ele possui o código aberto e este está disponibilizado em <https://github.com/gchq/CyberChef/>. O simulador pode ser utilizado em *browsers* pois está *online* no endereço <https://gchq.github.io/CyberChef/>. (GCHQ, 2020)

2.5.2 CrypTool

CrypTool é um portal de criptografia (<https://www.cryptool.org/en/>) que disponibiliza alguns simuladores de criptografia. Ele foi criado pela contribuição de várias instituições de vários locais do mundo, por exemplo: *University of Siegen* (Germany), *Politechnika Warszawska* (Poland), *Consejo Superior de Investigaciones Científicas* (Spain), *Klagenfurt University* (Austria) e *University of Singidunum* (Serbia). (CrypTool Contributors, 2020)

2.5.2.1 CrypTool 2

Figura 2 – Simulador: CrypTool 2

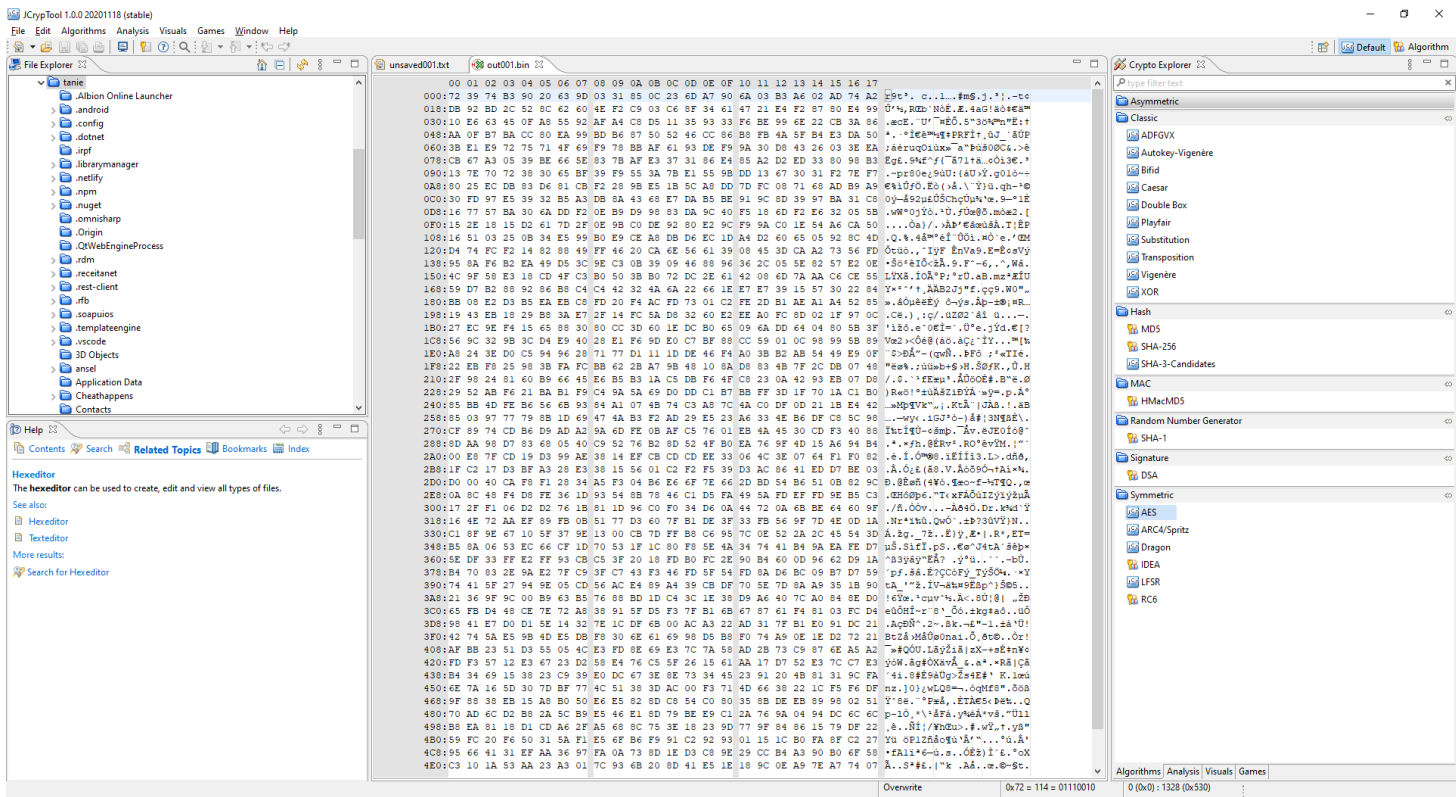


Fonte: do autor

O CrypTool 2 (Figura 2) está disponibilizado no idioma inglês e permite a utilização de vários algoritmos através da disponibilização de *templates* que podem ser criados pelos colaboradores. Nele pode-se informar os parâmetros de entrada que ele utiliza para mostrar o fluxo e resultado da execução do algoritmo. Ele é voltado ao ensino e também para testes dos algoritmos com entradas customizadas, dependendo do *template* escolhido. Só foi encontrado 1 *template* voltado ao ensino (a existência de *templates* voltados ao ensino dependem dos contribuidores). Ele possui o código aberto e este está disponibilizado em <https://www.cryptool.org/en/>. O simulador pode ser instalado somente no Windows e apresenta uma interface complexa e com muitas interações possíveis visíveis simultaneamente. (CrypTool Contributors, 2014)

2.5.2.2 JCrypTool

Figura 3 – Simulador: JCrypTool

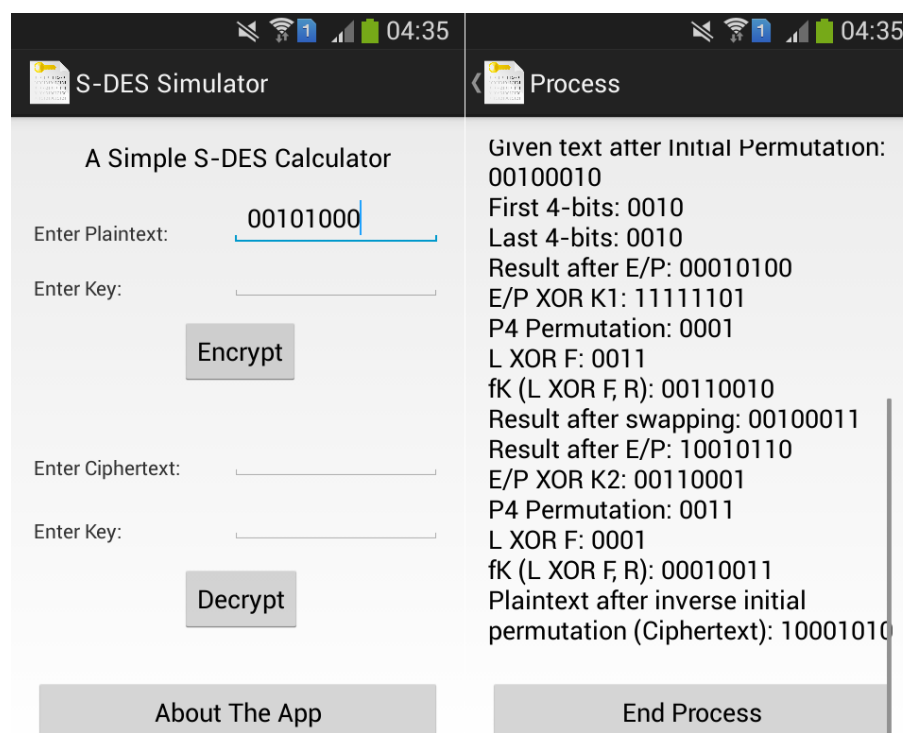


Fonte: do autor

O JCrypTool (Figura 3) está disponibilizado no idioma inglês, foi feito em java se utilizando como base a interface do *Eclipse* e permite a utilização de vários algoritmos através da disponibilização de *templates* que podem ser criados pelos colaboradores. Nele pode-se informar os parâmetros de entrada que ele utiliza para mostrar o resultado da execução do algoritmo. Ele é voltado ao ensino e para testes dos algoritmos com entradas customizadas, dependendo do *template* escolhido. Não foi encontrado *template* voltado ao ensino (a existência de *templates* voltados ao ensino dependem dos contribuidores). Ele possui o código aberto e este está disponibilizado em <https://www.cryptool.org/en/>. O simulador pode ser instalado em versões mais recentes do Windows, Linux e macOS e apresenta uma interface complexa e com muitas interações possíveis visíveis simultaneamente. (CrypTool Contributors, 2016)

2.5.3 S-DES Simulator (*Android App*)

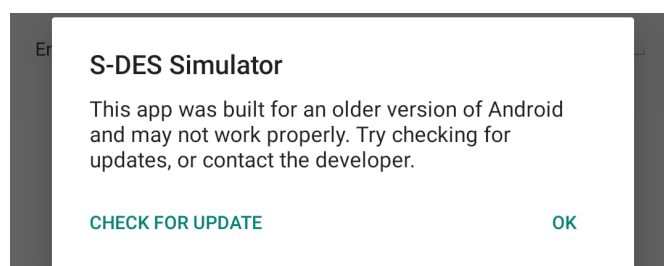
Figura 4 – Simulador: S-DES Simulator



Fonte: do autor

O S-DES Simulator (Figura 4) está disponibilizado no idioma inglês e permite a execução do S-DES, incluindo os resultados obtidos em cada passo. Nele pode-se informar os parâmetros de entrada que ele utiliza para mostrar os resultados dos passos e o resultado da execução do algoritmo. Ele é voltado ao ensino. O simulador foi desenvolvido para uma versão bem antiga do Android então precisa ser habilitado modo de compatibilidade na *App Store* e mesmo depois de instalado um aviso sempre aparece ao iniciar o aplicativo como mostra a figura 5. (MOUNTOGIANNAKIS, 2015)

Figura 5 – Simulador: S-DES Simulator's warning



Fonte: do autor

2.5.4 S-DES Simulator (*online*)

Figura 6 – Simulador: S-DES Simulator

S-DES

Plaintext	Key	Plaintext
<input type="text" value="test message"/>	<input type="text" value="274"/>	<input type="text" value="test message"/>
IP	P10	IP
<input type="text" value="11100100"/>	<input type="text" value="0010000101"/>	<input type="text" value="01100101"/>
f_k	Shift1	f_k
<input type="text" value="01000100"/>	<input type="text" value="0100001010"/>	<input type="text" value="11100100"/>
SW	K1	SW
<input type="text" value="01000100"/>	<input type="text" value="00100001"/>	<input type="text" value="01000100"/>
f_k	Shift2	f_k
<input type="text" value="11000100"/>	<input type="text" value="0000101001"/>	<input type="text" value="01000100"/>
IP	K2	IP
<input type="text" value="01000101"/>	<input type="text" value="00100110"/>	<input type="text" value="11000100"/>
Encryption		Encryption
<input type="text" value="»EH»\$ÉEHH E"/>		<input type="text" value="»EH»\$ÉEHH E"/>
<input type="button" value="RUN"/> <input type="button" value="REROAD"/>		

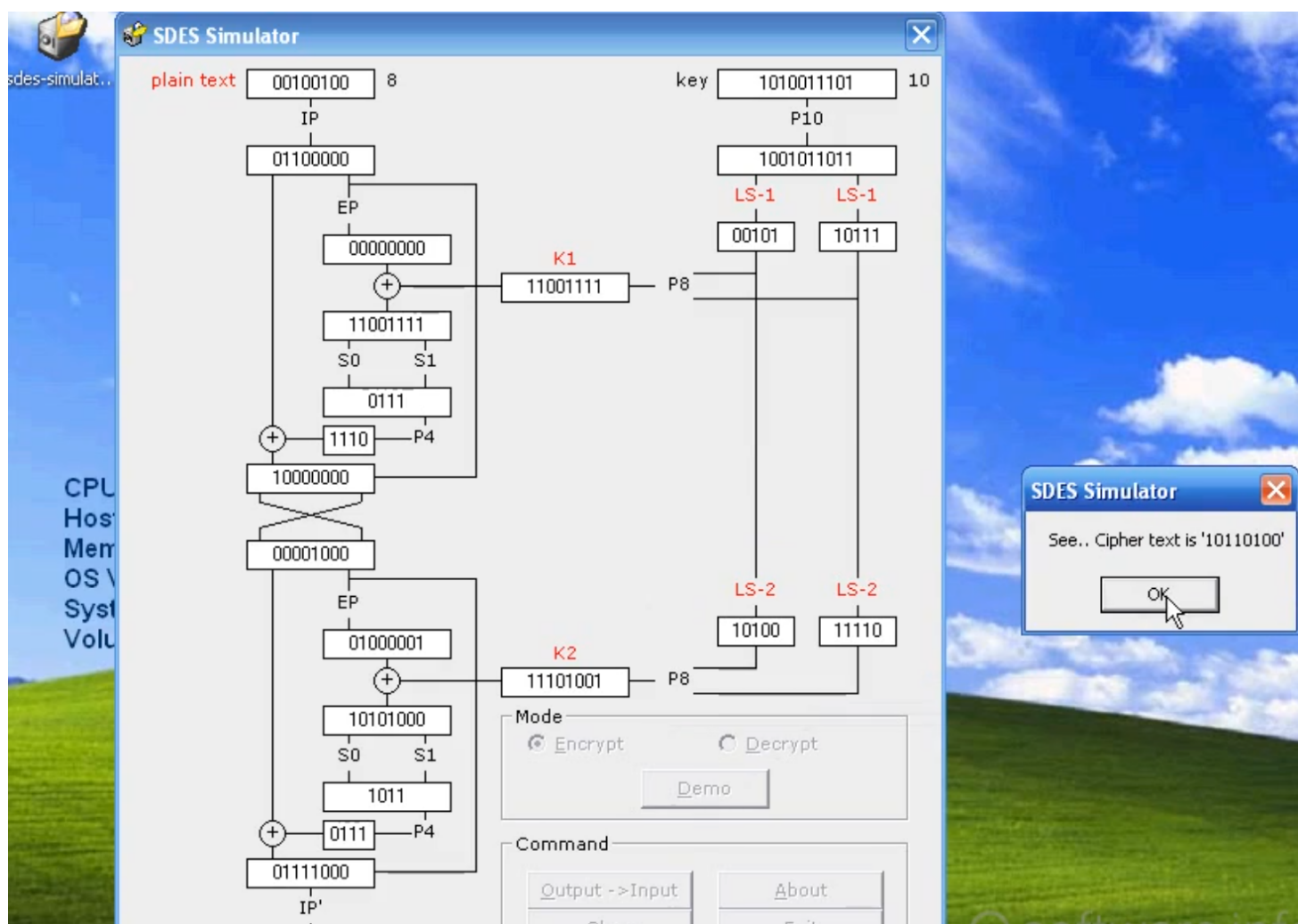
devgw | 설명서 [github](#)
 ES6(ES2015) 으로 작성되었으므로 Edge, Firefox, Chrom 에서 사용

Fonte: do autor

O S-DES Simulator (Figura 6) está disponibilizado no idioma coreano e permite a execução do S-DES, incluindo os resultados obtidos em cada passo. Nele pode-se informar os parâmetros de entrada que ele utiliza para mostrar os resultados dos passos e o resultado da execução do algoritmo. Ele é voltado ao ensino. Ele possui o código aberto e este está disponibilizado em <https://github.com/devGW/S-DES-simulator/>. O simulador pode ser utilizado em *browsers* pois está *online* no endereço <https://devgw.github.io/S-DES-simulator/>. (WOO, 2018)

2.5.5 S-DES Simulator (Win)

Figura 7 – Simulador: S-DES Simulator



Fonte: do autor

O S-DES Simulator (Figura 7) está disponibilizado no idioma inglês e permite a execução do S-DES, incluindo os resultados obtidos em cada passo. Nele pode-se informar os parâmetros de entrada que ele utiliza para mostrar os resultados dos passos e o resultado da execução do algoritmo. Ele é voltado ao ensino e pode ser instalado no Windows. (PERMADI, 2018)

2.6 Criptografia

Segundo a National Research Council (1991, apud STALLINGS, 2008, pg. 15), “A criptografia provavelmente é o aspecto mais importante da segurança de comunicações e está se tornando cada vez mais importante como um componente básico para a segurança do computador.”

O termo Criptografia vem do Grego *kryptós*, que significa “escondido” e de *gráphein*, que significa “escrita”. Ela é o estudo dos princípios e técnicas pelas quais os dados podem ser transformados da sua forma original em outra ilegível. Dessa forma, os dados podem ser conhecidos somente pelo destinatário, o detentor da “chave secreta”, o que faz com que mesmo que tenham sido interceptados, torne difícil a leitura de seu conteúdo por alguém não autorizado. Ela faz parte da Criptologia e é um sub-ramo da Matemática (KNUDSEN, 1998).

O estudo da maneira de camuflar o real significado de uma mensagem usando técnicas e algoritmos de cifragem têm evoluído juntamente com o estudo da maneira de se conseguir entender a mensagem quando não se é o real destinatário da mesma. Este campo de estudo é chamado Criptoanálise (GAINES, 1956). A Criptologia engloba a Criptografia e a Criptoanálise. Alguns autores se utilizam do termo Criptovirologia quando falam de vírus que se utilizam de chaves públicas (YOUNG; YUNG, 2004).

A criptografia se divide em Simétrica, também chamada de criptografia convencional, e Assimétrica, também chamada de criptografia por chave pública (STALLINGS, 2014) (TANENBAUM, 2003) e dentro dessa divisão ainda temos algoritmos de criptografia reversíveis e irreversíveis (STALLINGS, 2014) (ITU, 1991).

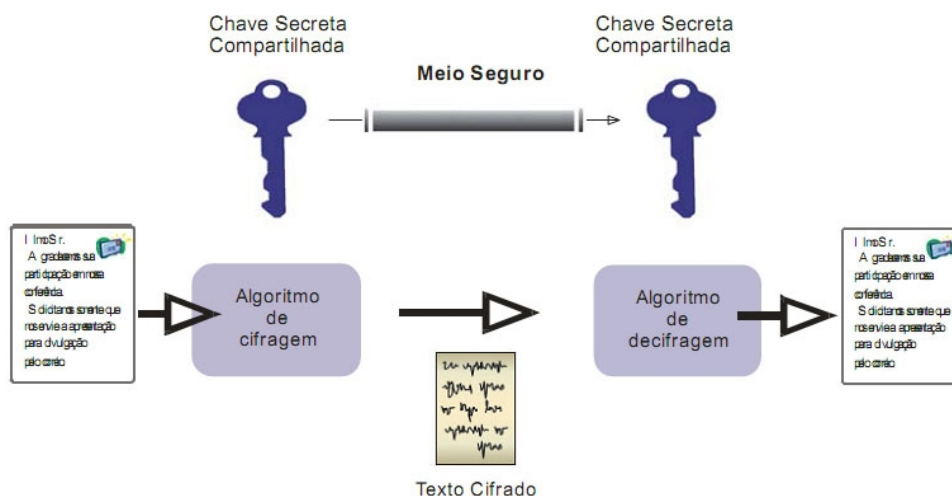
A criptografia possui inúmeras áreas de aplicação. Desde transações bancárias, envio de e-mails, segurança de arquivos sigilosos, segurança de autenticação, armazenamento de senhas em bancos de dados até o sinal de linha telefônica, sinal de TV digital e acesso a sites certificados (AVELINO; AVELINO, 2007).

2.6.1 Criptografia Simétrica

Também chamada de criptografia convencional, a criptografia simétrica é um criptosistema no qual tanto a criptografia como a deciptografia são realizadas com a mesma chave.

Ela transforma uma mensagem clara em uma mensagem cifrada, usando um algoritmo de criptografia e uma chave. Esta mesma chave juntamente com a inversão do algoritmo utilizado (gerando assim o algoritmo de deciptografia do algoritmo), são necessários para que se possa obter a mensagem clara novamente a partir da mensagem cifrada, assim como demonstra a Figura 8.

Figura 8 – Criptografia Simétrica



Fonte: (BROCARDO; ROLT; FERNANDES, 2006)

Antes do computador, as cifras simétricas tradicionais poderiam se utilizar de técnicas de transposição ou de substituição, ou até as duas técnicas combinadas. Essas técnicas são os componentes básicos para todas as técnicas de criptografia.

Técnicas de transposição transpõem sistematicamente as posições dos elementos da mensagem clara. Tal técnica consiste na aplicação de alguma permutação na mensagem clara de forma que a mensagem final seja ilegível, e somente o detentor da forma de como os elementos da mensagem foram permutados pode obter novamente a mensagem de forma clara. A cifra mais simples dessa técnica é a cifra de *Rail fence*. A Figura 9 ilustra como ficaria a crifa *Rail fence* de profundidade dois do texto “Técnica de transposição”.

Figura 9 – Exemplo de *Rail fence*

T c i a e r n p s ç o
é n c d t a s o i ã

Fonte: do autor

Técnicas de substituição mapeiam elementos, caracteres ou bits, da mensagem clara e as substitui por outras letras ou números ou até símbolos. Analisando a mensagem clara como sendo um conjunto de bits, então a substituição é definida pela troca de padrões de bits da mensagem clara por padrões de bits da mensagem cifrada. Como exemplo clássico temos a cifra de César, criada por Júlio César. A Figura 10 ilustra como ficaria a cifra de César com o texto “Tecnica de substituaçao”.

Figura 10 – Exemplo de cifra de César

Alfabeto:
Claro: a b c d e f g h i j k l m n o p q r s t u v w x y z
Cifra: t u v w x y z a b c d e f g h i j k l m n o p q r s

Mensagem clara: Tecnica de substituicao
Mensagem cifrada: Mxvgbvt wx lnu!mbmnbvth

Fonte: do autor

Existem basicamente dois os ataques possíveis em um algoritmo criptográfico. A criptoanálise, que se baseia nas características do próprio algoritmo de criptografia, e a força bruta, que engloba simplesmente repetidas tentativas de todas as chaves possíveis até encontrar a correta.

Antes da existência do computador existiam máquinas que implementavam a nível de hardware técnicas de substituição. Eram conhecidas como máquinas de rotor. Duas foram as máquinas de rotor mais conhecidas, a da Alemanha, conhecida como Enigma e a do Japão conhecida como Purple. Elas foram utilizadas durante a segunda guerra mundial e a quebra desses dois códigos pelos Aliados foi significativa para o resultado da guerra. Abaixo, a Figura 11 mostra a máquina Enigma com três rotores.

Figura 11 – Imagem da máquina Enigma



Máquina Enigma com três rotores, teclado, luzes e conexões para câmbio de codificação.
Fonte: Wikipedia. Fonte: (WIKIPÉDIA, 2020)

Algoritmos simétricos são utilizados em cenários em que a mensagem tem uma necessidade de ser decriptografada, por exemplo, o Kerberos, serviço de autenticação desenvolvido no *Massachusetts Institute of Technology* (MIT) como parte do projeto

Athena, que visa trazer segurança a cenários distribuídos abertos, se utiliza unicamente de cifra simétrica (STALLINGS, 2014) (TANENBAUM, 2003).

2.6.2 Criptografia Assimétrica

Também chamada de criptografia por chave pública, a criptografia assimétrica é um criptosistema onde a criptografia e a decryptografia são realizadas com chaves diferentes, uma pública e outra privada. Quando uma mensagem é criptografada com uma chave, somente a outra chave poderá decryptografar a mensagem. O criptosistema assimétrico mais utilizado é o RSA.

Ela transforma uma mensagem clara em uma mensagem cifrada utilizando uma das duas chaves acima citadas e um algoritmo de criptografia. E utilizando a outra chave citada acima e o algoritmo de decryptografia, é possível extrair a mensagem clara a partir da mensagem cifrada.

A criptografia assimétrica possui três utilizações básicas: confidencialidade, autenticação ou ambas.

A confidencialidade garante que somente o destinatário será capaz de ler a mensagem. Quando se tem essa necessidade, deve-se criptografar a mensagem com a chave pública do destinatário, enviar a mensagem criptografada para o destinatário, e ele, de posse da chave privada, poderá decryptografar a mensagem e dessa forma obter a mensagem original.

O fato do destinatário ter conseguido decryptografar a mensagem com a chave privada dele próprio garante que a mensagem só poderia ter sido decryptografada por ele, detentor da chave privada, mas não garante a identidade do remetente, visto que a chave utilizada para criptografar a mensagem é pública, assim como demonstra a Figura 12.

Figura 12 – Fluxo da mensagem criptografada quando o foco é confidencialidade



Fonte: (PIROPO, 2007)

A autenticidade garante que quem enviou a mensagem foi o remetente. Quando se tem essa necessidade, deve-se criptografar a mensagem com a chave privada do remetente,

enviar a mensagem criptografada para o destinatário, e ele, de posse da chave pública do remetente, poderá decriptografar a mensagem e assim obter a mensagem original.

O fato do destinatário ter conseguido decriptografar a mensagem com a chave pública do remetente comprova que a mensagem é realmente do remetente, mas não garante que o destinatário é o único que poderia decriptografar a mensagem, visto que qualquer pessoa pode possuir a chave pública do remetente, assim como demonstra a Figura 13.

Figura 13 – Fluxo da mensagem criptografada quando o foco é autenticidade

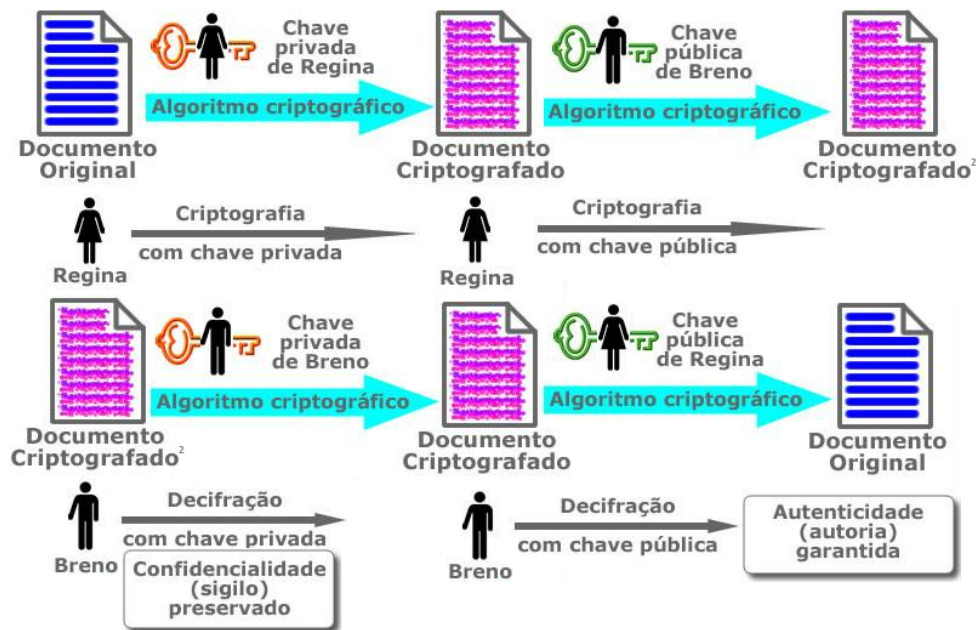


Fonte: (PIROPO, 2007)

É possível também que se criptografe a mensagem com a chave privada do remetente, criptografar a mensagem já criptografada com a chave pública do destinatário, enviar a mensagem duplamente criptografada para o destinatário, e ele de posse da chave privada, poderá decriptografar a mensagem duplamente criptografada em uma mensagem criptografada e esta por sua vez será decriptografada com a chave pública do remetente e finalmente obter a mensagem original.

A combinação das duas criptografias, a com chave privada do remetente e com a chave pública do receptor, garante tanto que quem vai receber a mensagem é realmente o destinatário correto como garante que o remetente é quem ele diz ser (STALLINGS, 2014) (TANENBAUM, 2003), assim como demonstra a Figura 14.

Figura 14 – Fluxo da mensagem criptografada quando o foco é confidencialidade e autenticidade

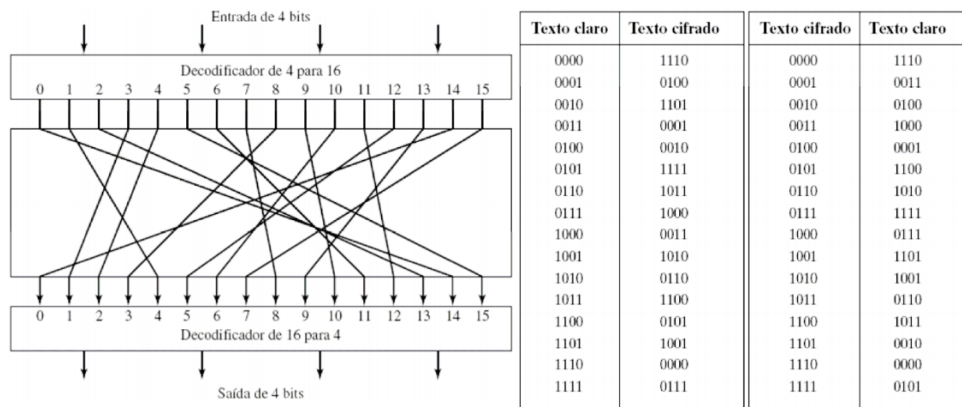


Fonte: (PIROPO, 2007)

2.7 Cifra de blocos

A cifra de blocos manipula um bloco da mensagem clara como um todo e utiliza este para criar um bloco de mensagem cifrada de mesmo tamanho. É comum a utilização de um bloco com 64 ou 128 bits. Uma cifra de bloco pode ser usada para alcançar o mesmo efeito que uma cifra de fluxo. Cifras de fluxo trata de certo fluxo de dados singularmente, bit a bit ou byte a byte. São implementações clássicas das cifras de fluxo as cifras de *Vigenère* auto chaveada e a cifra de Vernam e a mais utilizada atualmente é o RC4. A maioria das criptografias simétricas voltadas para ambientes de rede de computadores são implementadas com cifra de blocos e grande parte dos algoritmos de cifra de bloco simétrico usados na atualidade são baseados na cifra de blocos de *Feistel*.

Figura 15 – Cifra de bloco ideal



Fonte: (STALLINGS, 2010)

Uma cifra de substituição reversível qualquer, *Feistel* chamava esse cenário de cifra de bloco ideal, para um grande tamanho de bloco não é viável, do ponto de vista de desempenho e implementação, assim como demonstra a Figura 15. *Feistel* propunha que era possível chegar mais próximo da cifra de bloco ideal. Era só utilizar uma cifra de produto, ou seja, a execução de duas ou mais cifras em sequência, tornando assim o produto final criptograficamente mais forte do que se poderia obter através de qualquer uma das cifras componentes do produto. A estrutura de *Feistel* consiste em repetidas rodadas do mesmo procedimento. A cada vez que o procedimento se repete é realizada a substituição em metade dos dados da mensagem sendo processados, logo após se permuta as duas metades. A chave sendo utilizada é expandida de modo que uma chave diferente seja utilizada em cada iteração. Na prática *Feistel* propôs o uso de uma cifra que alternava entre substituições e permutações, o que na realidade é uma implementação prática do que *Claude Shannon* já havia proposto como, cifra de produto que alterne confusão e difusão (STALLINGS, 2014) (TANENBAUM, 2003).

2.8 DES

Foi estabelecido pela IBM no final da década de 1960 um projeto de pesquisa sobre criptografia de computadores que trazia a sua frente *Horst Feistel*. Com a conclusão do projeto em 1971 foi apresentado como resultado um algoritmo denominado LUCIFER, que foi comercializado ao *Lloyd's* localizado em Londres para ser utilizado em um sistema de caixa eletrônico que também havia sido desenvolvido pela IBM. LUCIFER é uma cifra de bloco de *Feistel* que operava com blocos de 64 bits e se utilizava de uma palavra com tamanho de 128 bits.

Com o sucesso do LUCIFER a IBM mobilizou um novo projeto, dessa vez com o intuito de tornar o LUCIFER comercializável em grande escala, conseguindo coloca-lo em

um único chip. Este projeto, que por sua vez era liderado por *Walter Tuchman* e *Carl Meyer*, contava com a ajuda tanto de consultores externos como orientação técnica da *National Security Agency* (NSA). O resultado deste novo projeto foi um LUCIFER mais refinado e mais resistente à criptoanálise, mas continha um tamanho de chave de 56 bits, para poder caber em um chip.

A *National Bureau of Standards* (NBS), em 1973, solicitou propostas de uma cifra para ser padronizada a nível nacional. A IBM enviou o LUCIFER refinado, o que continha 56 bits, e por ser o melhor algoritmo proposto foi, em 1977, adotado como *Data Encryption Standard* (DES).

Com uma chave de 56 bits, existem ao todo 256 possíveis chaves, algo próximo de $7,2 \times 10^{16}$ chaves. Aparentemente, um número tão grande de possibilidades é algo praticamente impossível de se descobrir se utilizando o método de força bruta. Por exemplo, uma única máquina processando uma criptografia DES por microssegundo levaria mais de mil anos para quebrar a cifra.

Por mais que para as máquinas de hoje, uma criptografia DES a cada microssegundo pareça irreal, pois a velocidade das máquinas de hoje já é bem superior as máquinas da década de 70, para aquela época não era algo tão simples. Mesmo assim, em 1977, *Diffie* e *Hellman* declararam que já existia tecnologia suficiente para se criar uma máquina que concentraria, de forma paralela, um milhão de dispositivos criptográficos, cada um com o poder de processamento da máquina citada no exemplo anterior. Isso reduziria o tempo de quebra da cifra para cerca de dez horas. Infelizmente, tal configuração na época custaria em torno de vinte milhões de dólares.

Finalmente, em 1998, vinte e um anos depois, o DES provou ser inseguro, quando a *Electronic Frontier Foundation* (EFF) anunciou que tinha quebrado uma cifra DES utilizando uma máquina “decifradora de DES” montada por menos de 250 mil dólares. Como se isso já não fosse suficiente para tornar o DES um algoritmo criptográfico ultrapassado, ainda existe a constante evolução do hardware, aumentando cada vez mais a velocidade de processamento e consequentemente possibilitando tanto que o método da força bruta seja mais viável, temporalmente falando, como também que sejam criados algoritmos de criptografia que se utilizem melhor desse novo poder de processamento.

Felizmente já existem várias alternativas para o DES, entre elas estão o *Advanced Encryption Standard* (AES) e o *Triple Data Encryption Standard* (3DES) (STALLINGS, 2014) (TANENBAUM, 2003).

2.9 3DES

A criptografia múltipla é quando um algoritmo de criptografia é utilizado repetidas vezes. Primeiramente, a mensagem clara é criptografada utilizando um algoritmo de criptografia. A mensagem cifrada é então usada como entrada para um algoritmo de criptografia, podendo ser o mesmo utilizado anteriormente ou algum outro algoritmo, e esse processo pode ser repetir por indefinidas vezes.

O *Triple Data Encryption Standard* (3DES) é um exemplo de criptografia múltipla. Ele se utiliza do algoritmo DES três vezes, usando duas ou três chaves diferentes.

Em seu estado inicial a criptografia múltipla possui dois estágios, no caso do DES duplo, cada um se utilizando de uma chave diferente uma da outra, a criptografia E de uma mensagem M se utilizando de duas chaves K_1 e K_2 resultando em um texto criptografado C seria assim:

$$C = E(K_2, E(K_1, M))$$

Já a sua decryptografia D seria assim, aplicando as chaves inversamente:

$$M = D(K_1, D(K_2, C))$$

O DES triplo veio como uma alternativa clara para sanar o problema do DES simples, gerado pelo avanço computacional, uma vez que aumenta o custo do ataque da mensagem clara conhecida para 2112 (quando se utiliza de duas chaves) o que está além do possível, pelo menos, atualmente. Mas se utilizar de três estágios com três chaves diferentes exige um tamanho de chave de 168 bits (56×3), o que pode ser custoso.

Para diminuir o problema citado, *Tuchman* propôs uma alternativa se utilizando somente de duas chaves:

$$C = E(K_1, D(K_2, E(K_1, M)))$$

A solução acima apresentada se tornou relativamente popular tendo sido adotada para uso nos padrões de gerenciamento de chaves ANS X9.17 e ISO 8732.

Atualmente contra o 3DES não existem ataques criptoanalíticos práticos, visto que, o custo de uma pesquisa de chave por força bruta seria de uma complexidade de ordem $2^{56} \times 2^{56} = 2^{112}$ (quando fossem utilizadas somente duas chaves) o que é equivalente a aproximadamente 5×10^{33} .

Mesmo o 3DES com duas chaves sendo de difícil ataque, ainda pode haver uma certa preocupação. Portanto pesquisadores creem que o melhor seria se utilizar do 3DES

com três chaves. Como já dito anteriormente o 3DES com três chaves possui uma chave de tamanho efetivo de 168 bits, o que a torna mais segura em relação com o de duas chaves, e possui a seguinte forma (STALLINGS, 2014):

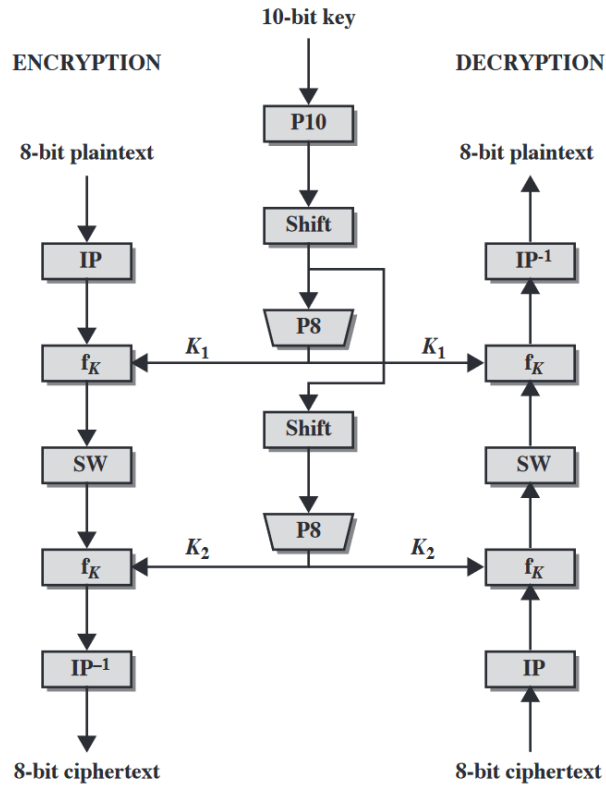
$$C = E(K_3, D(K_2, E(K_1, M)))$$

2.10 S-DES

O *Simplified Data Encryption Standard* (S-DES), desenvolvido pelo professor *Edward Schaefer* da *Santa Clara University* (SCHAEFER, 1996), é uma versão simplificada e voltada ao ensino do algoritmo *Data Encryption Standard* (DES) do qual possui as mesmas propriedades e estrutura tendo como únicas divergências o tamanho reduzido dos parâmetros de entrada e o número reduzido de execuções da função f_K . Enquanto o DES recebe como entrada blocos de 64 bits, usa 1 chave de 56 bits de onde se extraem 16 chaves de 48 bits (cada uma será utilizada em uma das 16 aplicações de f_K) e retorna como saída blocos de 64 bits o S-DES recebe como entrada 1 bloco de 8 bits, usa 1 chave de 10 bits de onde se extraem 2 chaves de 8 bits (cada uma será utilizada em uma das 2 aplicações de f_K) e retorna como saída 1 bloco de 8 bits. Essas reduções tornam o S-DES o melhor candidato para análise quando o objetivo é aprendizado. (STALLINGS, 2010) (STALLINGS, 2014)

A figura 16 apresenta a estrutura geral do S-DES. Nela pode-se observar quais são as etapas que geram as chaves, quais etapas e ordem destas criptografam o *plaintext* e quais etapas e ordem destas descriptografam o *ciphertext*.

Figura 16 – Estrutura do S-DES



Fonte: "Figura G.1 do apêndice G"(STALLINGS, 2010)

Pode-se expressar matematicamente o algoritmo de criptografia como uma composição de funções:

$$IP^{-1} \circ f_{K_2} \circ SW \circ f_{K_1} \circ IP$$

Ou de maneira mais direta:

$$ciphertext = IP^{-1}(f_{K_2}(SW(f_{K_1}(IP(plaintext)))))$$

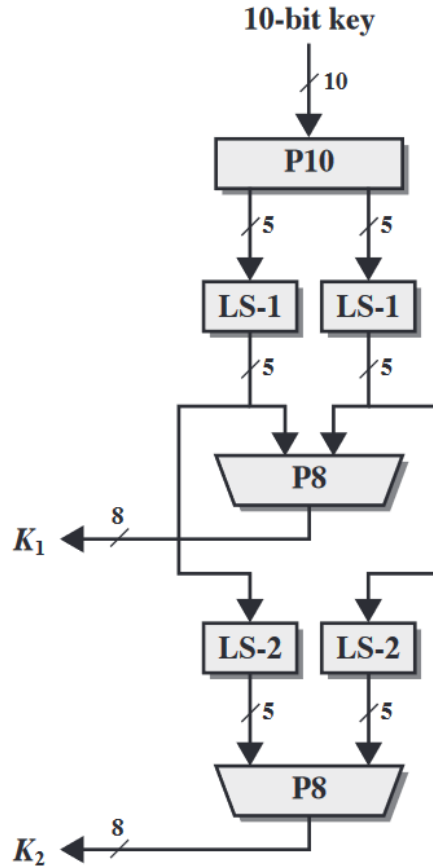
Similarmente pode-se expressar o algoritmo de descryptografia:

$$IP^{-1} \circ f_{K_1} \circ SW \circ f_{K_2} \circ IP$$

Ou também:

$$plaintext = IP^{-1}(f_{K_1}(SW(f_{K_2}(IP(ciphertext)))))$$

Figura 17 – Geração das Chaves do S-DES



Fonte: "Figura G.2 do apêndice G"(STALLINGS, 2010)

A geração das chaves pode ser expressada através das seguintes:

$$K_1 = P8(Shift_1(P10(key)))$$

$$K_2 = P8(Shift_2(Shift_1(P10(key))))$$

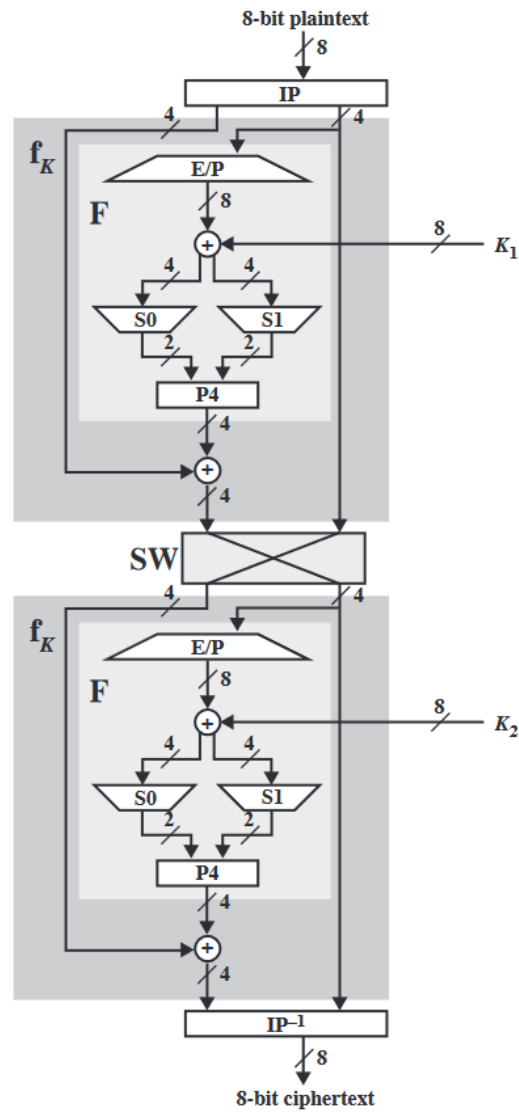
A figura 18 demonstra de forma mais detalha o fluxo de execução de criptografia do S-DES. Nesta a função f_K é detalhada. Esta recebe por sua vez 3 parâmetros: L , R e K . L é a metade esquerda dos bits recebidos do passo anterior, R são os bits restantes, ou seja, a metade direita e K é a *key* que deve ser utilizada. Sendo assim, temos:

$$f_K(L, R) = (L \oplus F(R, K), R)$$

A função F por sua vez não é tão diretamente explícita matematicamente mas seu fluxo também pode ser analisado na figura 18. (STALLINGS, 2010)

A descrição detalhada completa do fluxo de execução do S-DES pode ser encontrada no simulador desenvolvido.

Figura 18 – Detalhamento da criptografia do S-DES



Fonte: "Figura G.3 do apêndice G"(STALLINGS, 2010)

3 O simulador

Essa seção descreve o **CryptoEdu - simulador de algoritmos de criptografia com finalidade educacional** criado no decorrer da escrita desse trabalho de conclusão de curso.

O simulador permite a execução completa e passo a passo de todas as etapas envolvidas tanto na criptografia quando na descriptografia utilizando o algoritmo *Simplified Data Encryption Standard* (S-DES), escolhido por ser um algoritmo mais recomendado para ensino, como explicado na seção 2.10.

O público alvo da ferramenta é o corpo docente e discente de cursos de TI. Mas também, pode ser utilizada por todos que desejam aprender como a criptografia de cifra de blocos funcionada, refinar e/ou lapidar os conhecimentos já adquiridos ou até somente desmistificar esse conteúdo complexo chamado criptografia.

3.1 Informações técnicas

O simulador foi desenvolvido na linguagem *Javascript* utilizando *React.js* e *Typescript*. O *framework* de interface utilizado foi o *Material-UI* disponibilizado no endereço <https://material-ui.com/>. O ambiente de desenvolvimento utilizado foi o *Visual Studio Code*. O código fonte do simulador é *open source* e está disponibilizado no *GitHub* no endereço <https://github.com/TanielianVB/CryptoEdu/>.

A maior parte dos componentes utilizados são nativos do próprio *framework de interface*. O simulador no entanto possui 3 tipos de componentes: 1. Componentes de customização, criados para facilitar a utilização de componentes nativos do *framework*, sem duplicidade de código.; 2. Componentes de agregação, criados para facilitar a utilização de vários componentes simultaneamente.; 3. Componentes de funcionalidade, criados para encapsular uma lógica. Deste último tipo pode-se citar os executores bit a bit que podem inclusive ser utilizados por outros algoritmos no futuro.

No *GitHub* é possível: Gerenciar problemas encontrados e melhorias sugeridas através de *Issues*.; Integrar melhorias implementados por terceiros através de *Pull Requests*.; A publicação de qualquer nova versão está automatizada através da ferramenta *Netlify* (<https://www.netlify.com/>) e reagindo à qualquer nova versão submetida ao repositório. Esses *deploys* podem ser visualizados em <https://app.netlify.com/sites/cryptoedu/deploys/>.; E até, caso necessário, criar uma nova versão do simulador de propriedade de terceiros através do *Fork*.

O simulador está disponibilizado ao usuário através da internet no endereço

<https://cryptoedu.netlify.app/>. O simulador pode ser visualizado em *browsers*, inclusive em dispositivos móveis. No entanto a melhor experiência é obtida quando se utiliza uma resolução maior. Normalmente só obtida se utilizando *desktops*. Embora não seja o ideal, o simulador se comporta bem em dispositivos móveis melhorando ainda mais a sua acessibilidade (figura 19).

Figura 19 – Visualização em um *browser* de dispositivo móvel

CryptoEdu S-DES

Simulador de Algoritmos de Criptografia com Finalidade Educacional

S-DES - Simplified Data Encryption Standard

CRIPTOGRAFAR DESCRIPTOGRAFAR

Mensagem *
01110010
1 char ou 8 bits
Bits da mensagem:

1	2	3	4	5	6	7	8	Char
0	1	1	1	0	0	1	0	r

Chave *
1010000010
10 bits
Bits da chave:

1	2	3	4	5	6	7	8	9	10
1	0	1	0	0	0	0	0	1	0

☐ Iniciar regiões de explicação abertas

PRÓXIMO >

1 — 2 — 3 — 4 — 5 — 6 — 7 — 8

I C IP f_{K_1} SW f_{K_2} IP^{-1} F

Fonte: do autor

A responsividade foi implementada com um misto de reposicionamento e redimensionamento levando em consideração *breakpoints*. Ou seja, baseado no tamanho visível da tela no momento os componentes vão se redimensionar ou se reposicionar. Os *breakpoints* considerados (figura 20) foram: *extra-small*, de 0px à 600px; *small*, de 601px à 960px; *medium*, de 961px à 1280; *large*, de 1281px à 1920px; e *extra-large*, acima de 1920px. Por exemplo, na tela inicial os campos **Mensagem** e **Chave** ficarão um ao lado do outro (figura 29) até o *breakpoint medium* (961px). No entanto, ao entrar no *breakpoint small*

(960px) estes campos ficarão um sobre o outro (figura 19).

Figura 20 – *UI Breakpoints*

value	0px	600px	960px	1280px	1920px
key	xs	sm	md	lg	xl
screen width	----- ----- ----- ----- ----->				
range	xs	sm	md	lg	xl

Fonte: do autor

O simulador foi concebido prevendo uma fácil extensão das suas atuais funcionalidades. Mas, visto que o escopo do projeto pode se exceder além do limite possível de execução de um trabalho de conclusão de curso, algumas decisões foram tomadas para viabilizar o desenvolvimento da ferramenta em tempo hábil. São elas:

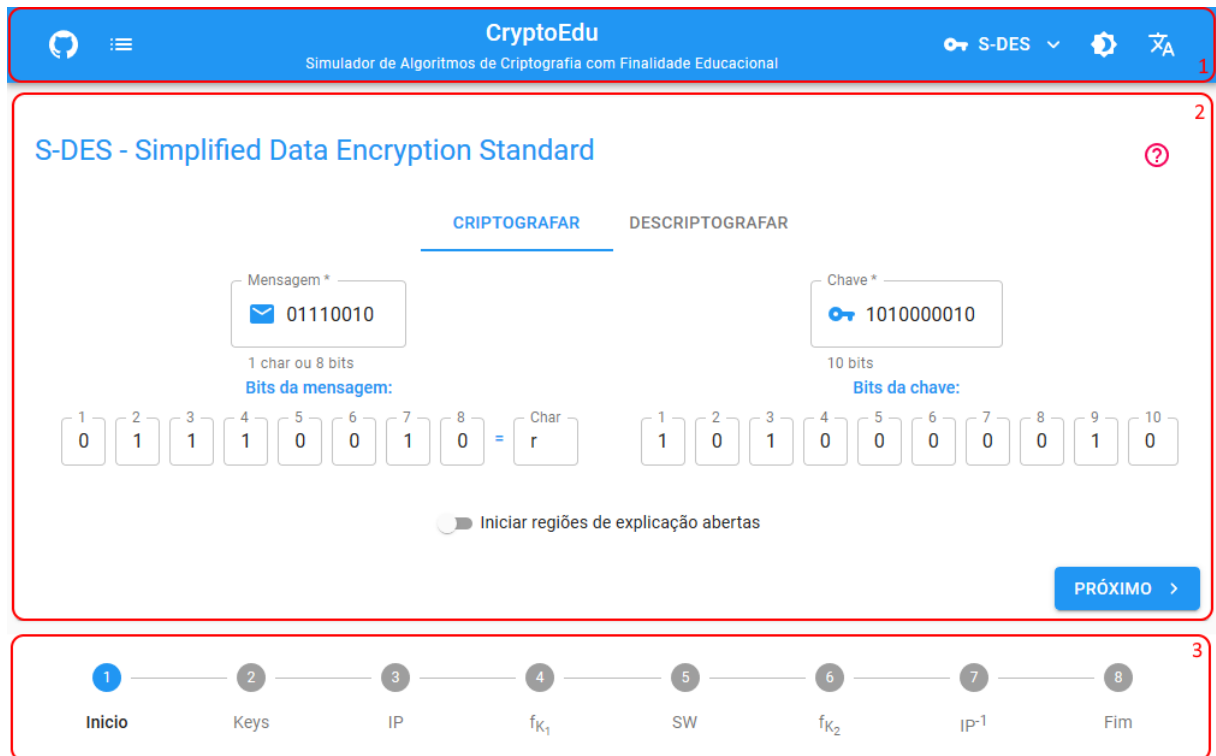
- Somente 1 algoritmo será disponibilizado. Sendo este, o *Simplified Data Encryption Standard* (S-DES).
- Só estará disponibilizado no tema **claro**.
- Só estará disponibilizado em **Português-BR**.
- A interface será otimizada para dispositivos *desktop* mas também irá poder ser visualizada em dispositivos *mobile*.

3.2 Estrutura da interface

Nessa seção são descritos os elementos contidos na interface de maneira estrutural.

A interface do simulador é composta por 3 seções principais, como pode ser visto na figura 21: 1. cabeçalho; 2. conteúdo; e 3. rodapé. Descritas nas sub-seções abaixo.

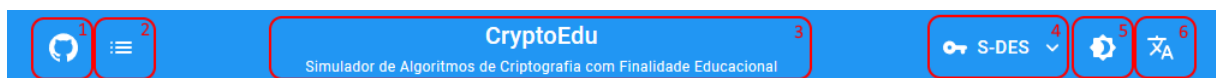
Figura 21 – Estrutura da interface



Fonte: do autor

3.2.1 Cabeçalho

Figura 22 – Cabeçalho



Fonte: do autor

A região do cabeçalho (figura 22) possui:

1. Link do repositório onde está localizado tanto o código fonte do simulador como um pdf do TCC: <https://github.com/TanielianVB/CryptoEdu>
2. Link do questionário de *feedback* de utilização do simulador.
3. O nome do simulador: **CryptoEdu - Simulador de Algoritmos de Criptografia com Finalidade Educacional**.
4. *Combo Box* de escolha do algoritmo que está em execução.
5. Botão para alternar entre o tema **claro** e **escuro**.
6. Botão para alterar o idioma no qual o simulador está sendo exibido.

3.2.2 Conteúdo

Figura 23 – Conteúdo

Fonte: do autor

A região principal da página será onde cada passo da execução do algoritmo selecionado irá ocorrer. À cada passo serão exibidos nessa região as etapas contidas nesse passo. Inicialmente irá a interface do passo inicial (figura 23). É possível navegar pelos passos através dos botões de navegação (Próximo, Anterior e Reiniciar) exibidos na parte inferior da região de cada passo. No primeiro passo o botão Anterior não é exibido, somente nos outros. No último passo o botão Próximo é substituído pelo botão Reiniciar para facilitar o início de uma nova execução, como mostra a figura 24.

Figura 24 – Navegação no último passo

Fonte: do autor

3.2.3 Rodapé

Figura 25 – Rodapé



Fonte: do autor

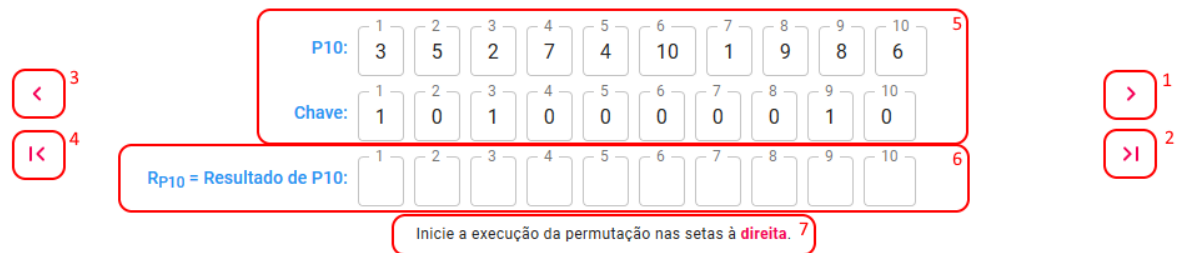
A região do rodapé (figura 25) contém:

1. Todos os passos necessários para a execução do algoritmo em execução. Ao se passar o mouse sobre eles é exibida uma *tooltip* com um nome mais explicativo do passo.
2. Quais passos já foram completados.
3. Qual passo o usuário se encontra no momento.
4. Quais passos ainda faltam ser completados para finalizar a execução do algoritmo.

É possível também, a qualquer momento, navegar para qualquer passo diretamente. Basta-se clicar sobre este.

3.2.4 Executor bit a bit

Figura 26 – Executor bit a bit no início da execução da etapa



Fonte: do autor

Um componente de interface muito utilizado é o executor **bit a bit** que permite que o usuário visualize a execução da etapa de maneira granular e explicativa. Este componente, como mostra a figura 26, é composto por:

1. Vai para o próximo passo da execução da etapa.
2. Vai para o último passo da execução da etapa.
3. Vai para o passo anterior da execução da etapa.
4. Vai para o primeiro passo da execução da etapa.
5. Entradas recebidas para execução da etapa.
6. Saída da etapa, que é construída a cada passo da execução desta.
7. Explicação do passo da execução da etapa sendo exibido no momento.

Figura 27 – Executor bit a bit durante a execução da etapa



Fonte: do autor

Durante a execução da etapa, os bits envolvidos nesta são realçados, o resultado vai sendo preenchido e a explicação na parte de baixo do executor vai se ajustando ao contexto sendo apresentado pelo executor, como mostra a figura 27.

Figura 28 – Executor bit a bit no fim da execução da etapa



Fonte: do autor

Ao fim da execução da etapa, o resultado vai estar todo preenchido e a explicação na parte de baixo do executor irá indicar o resultado da etapa, como mostra a figura 28.

3.3 Interface do conteúdo

Considera-se passo como sendo a sequência lógica da execução do algoritmo S-DES como modelada pelo seu autor (figura 16), exemplo: Geração das chaves, Permutação inicial, etc.. Considera-se etapa como sendo a sequência de subprocessos executada dentro de um passo, exemplo: P10, LS-1, etc.. Considera-se um processo como sendo a operação computacional utilizada por uma etapa, exemplo: P10 e P8 são permutações.

Cada passo possui no mínimo uma etapa e cada etapa possui uma região de explicação onde a etapa é descrita de maneira que possibilite o usuário compreender como a etapa é executada, como essa deve ser compreendida, como esta difere de outras etapas similares e qual a definição matemática desta etapa, caso haja.

A explicação pode ser acessada através do clique do botão interrogação à direita do cabeçalho que contém o título da referida etapa (figura 23 elemento 1). Esse clique

irá expandir a região que contém a explicação da etapa, como pode ser visto na figura 29 elemento 1. É possível também que todas as regiões de explicação já iniciem abertas. Basta habilitar a opção **Iniciar regiões de explicação abertas** (figura 29 elemento 2) ao iniciar a execução da criptografia ou descryptografia.

Figura 29 – Explicação expandida

S-DES - Simplified Data Encryption Standard 1

O S-DES é uma versão simplificada do algoritmo DES (Data Encryption Standard). Aquele se utiliza de uma chave de 10 bits que deve ser compartilhada entre o emissor e o receptor da mensagem para que a mensagem possa ser criptografada e descryptografada.

Nesta execução (que possui objetivo educacional), podemos escolher se desejamos criptografar ou descryptografar a mensagem e informar uma mensagem e uma chave que irão ser utilizadas durante a execução do algoritmo para que, assim, consigamos melhor visualizar como ocorre o processo quando os valores colocados são utilizados.

CRIPTOGRAFAR
DESCRIPTOGRAFAR

Mensagem * 01110010

1 char ou 8 bits

Bits da mensagem:

1
0

2
1

3
1

4
1

5
0

6
0

7
1

8
0

=

Char
r

Chave * 1010000010

10 bits

Bits da chave:

1
1

2
0

3
1

4
0

5
0

6
0

7
0

8
0

9
1

10
0

☒ Iniciar regiões de explicação abertas 2

PRÓXIMO >

Fonte: do autor

3.3.1 Tela inicial - entrada de dados

A interface inicial (figura 29) descreve o algoritmo selecionado na *Combo Box* de escolha de algoritmo (na região de explicação) buscando situar o usuário no contexto selecionado para execução. O usuário pode então escolher se ele deseja o fluxo de execução **Criptografar** ou **Descryptografar** e assim então informar valores customizados para os campos **Mensagem** (**Mensagem cifrada** caso o fluxo escolhido tenha sido o **descryptografar**) e **Chave**.

Os algoritmos recebem como entrada bits. Por conta disso, são exibidos os bits contidos nos campos e que serão utilizados para a execução do fluxo escolhido. Tanto para facilitar o preenchimento do campo **Mensagem** como para melhorar a compreensão do usuário sobre o valor contido no campo, é possível informar no campo uma letra, visto que essa pode, e é, convertida para 8 bits. Isso ajuda não somente no preenchimento do campo durante a execução do fluxo **criptografar** mas também na validação do resultado obtido da execução do fluxo **descryptografar** visto que é possível, mais facilmente, comparar

as mensagens (tanto a de entrada do fluxo **criptografar** quando a de saída do fluxo **descriptografar**) se estas forem letras. Exemplo: A letra 'T' gera a sequência de bits: **01010100**.

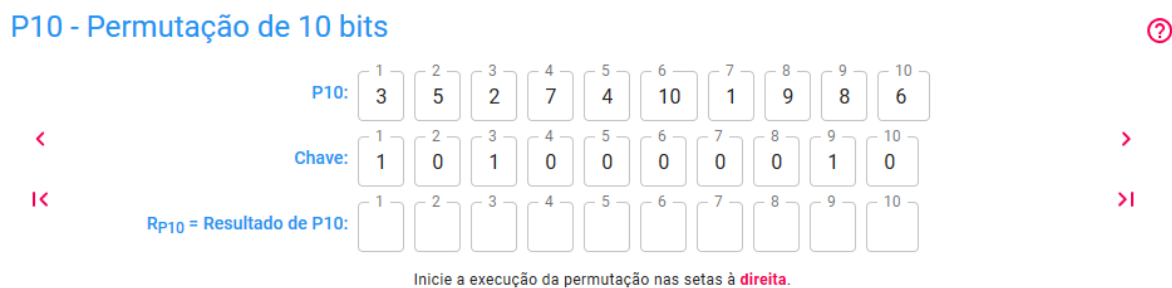
Nessa tela também é possível habilitar a opção **Iniciar regiões de explicação abertas** que fará com que, durante a navegação pelos passos da execução do algoritmo todas as regiões de explicação já se iniciem abertas. Caso o usuário não queria mais visualizar a explicação para alguma etapa. Ele pode esconder a explicação clicando no botão à direita do título da etapa.

3.3.2 Passo Chaves - Geração das chaves

Nesse passo estão concentradas todas as etapas necessárias para a geração das chaves K_1 e K_2 . As etapas contidas nesse passo estavam inicialmente divididas em 3 passos. Mas por ter gerado dúvida sobre como essas etapas se relacionavam tanto entre elas como com os outros passos da execução do algoritmo, estas etapas estão em um mesmo passo.

3.3.2.1 Etapa P10 - Permutação de 10 bits

Figura 30 – Etapa P10 - Permutação de 10 bits

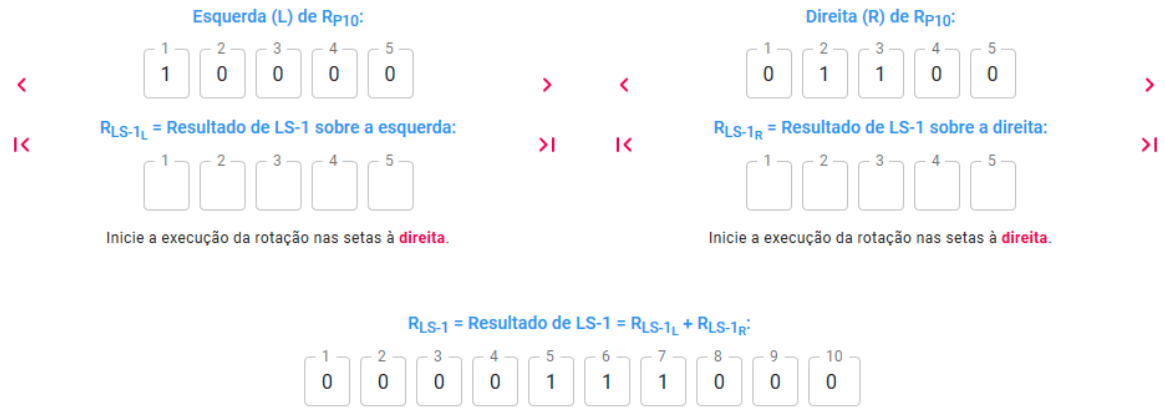


Fonte: do autor

A interface da etapa P10 (figura 30), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação P10 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exhibe um componente capaz de executar passo a passo a função de permutação P10. O parâmetro de entrada da permutação P10 é a Chave recebida da tela de entrada de dados. O resultado dessa execução será o parâmetro de entrada para a próxima etapa, LS-1.

3.3.2.2 Etapa LS-1 - *Circular Left Shift* de 1 posiçãoFigura 31 – Etapa LS-1 - *Circular Left Shift* de 1 posição

LS-1 - Circular Left Shift de 1 posição



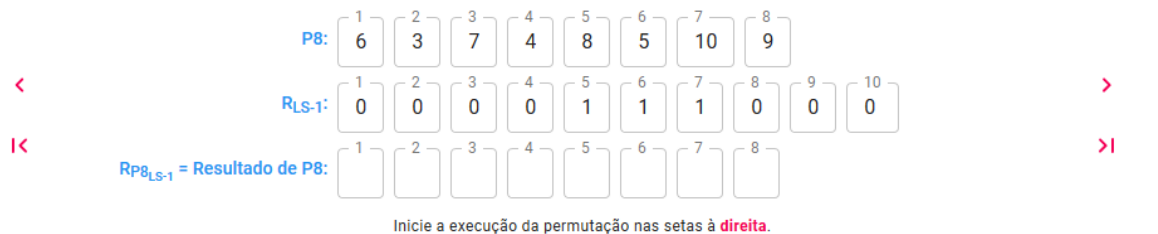
Fonte: do autor

A interface da etapa LS-1 (figura 31), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe e explica como esta etapa deve ocorrer, incluindo a divisão na metade do valor obtido na etapa anterior (P_{10}) e o que é o processo de rotação. Após tal contextualização se exibe dois componentes capazes de executar passo a passo a rotação circular para a esquerda de 1 posição. Cada um destes componentes irá rotacionar uma das metades. O resultado desta etapa é então a junção das metades após suas rotações individuais.

3.3.2.3 Etapa P8 - Permutação de 8 bits sobre o resultado de LS-1

Figura 32 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-1

P8 - Permutação de 8 bits



Fonte: do autor

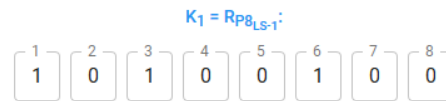
A interface da etapa P8 (figura 32), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta

a função de permutação P8 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação P8. O parâmetro de entrada da permutação P8 é o resultado obtido na etapa anterior, LS-1.

3.3.2.4 Resultado Chave K_1

Figura 33 – Chave K_1 - Geração da primeira chave K_1

K_1 - Primeira chave



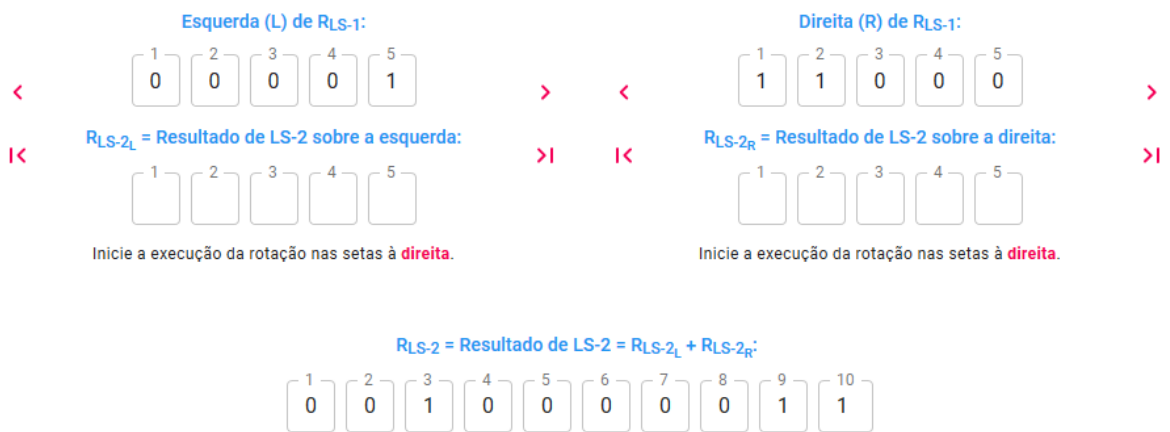
Fonte: do autor

O resultado da aplicação da função de permutação P8 sobre o resultado da etapa LS-1 será a primeira chave K_1 (figura 33). Esta será utilizada no primeiro passo f_K durante a criptografia ou no segundo passo f_K durante a descryptografia.

3.3.2.5 Etapa LS-2 - *Circular Left Shift* de 2 posições

Figura 34 – Etapa LS-2 - *Circular Left Shift* de 2 posições

LS-2 - Circular Left Shift de 2 posições



Fonte: do autor

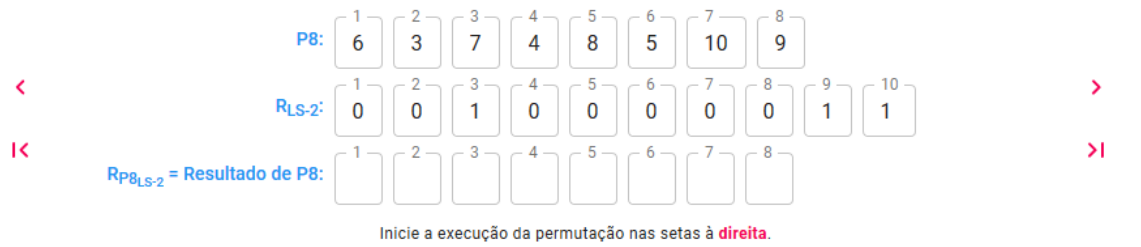
A interface da etapa LS-2 (figura 34), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe e explica como esta etapa deve ocorrer, incluindo a divisão na metade do valor obtido na etapa LS-1 e o que é o processo de rotação. Após tal contextualização se exibe dois componentes capazes de executar passo a passo a rotação

circular para a esquerda de 2 posições. Cada um destes componentes irá rotacionar uma das metades. O resultado desta etapa é então a junção das metades após suas rotações individuais.

3.3.2.6 Etapa P8 - Permutação de 8 bits sobre o resultado de LS-2

Figura 35 – Passo P8 - Permutação de 8 bits sobre o resultado de LS-2

P8 - Permutação de 8 bits



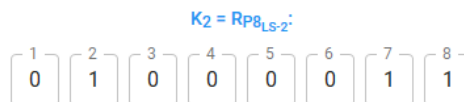
Fonte: do autor

A interface da etapa P8 (figura 35), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação P8 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação P8. O parâmetro de entrada da permutação P8 é o resultado obtido na etapa anterior, LS-2.

3.3.2.7 Resultado Chave K_2

Figura 36 – Chave K_2 - Geração da segunda chave K_2

K_2 - Segunda chave

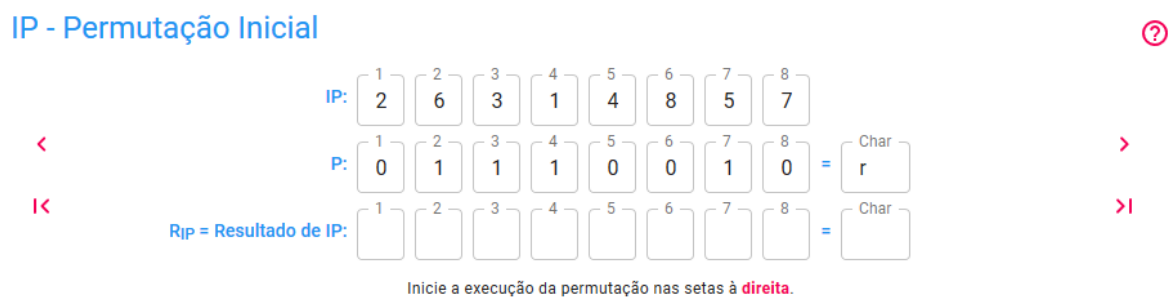


Fonte: do autor

O resultado da aplicação da função de permutação P8 sobre o resultado da etapa LS-2 será a segunda chave K_2 (figura 36). Esta será utilizada no segundo passo f_K durante a criptografia ou no primeiro passo f_K durante a descriptografia.

3.3.3 Passo IP - Permutação Inicial

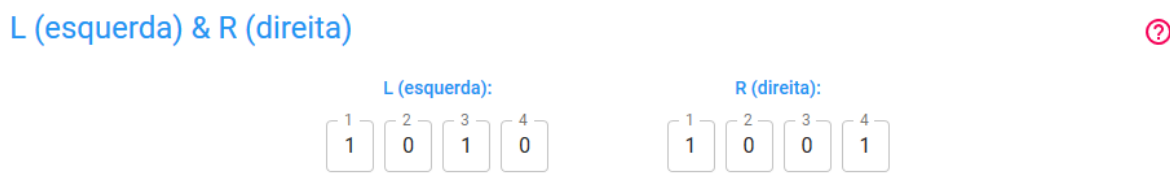
Figura 37 – Passo IP (*Initial Permutation*) - Permutação Inicial



Fonte: do autor

A interface da etapa IP (figura 37), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação IP (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação IP. O parâmetro de entrada da permutação IP é a Mensagem (*Plaintext* P) recebida da tela de entrada de dados.

Figura 38 – L (esquerda) & R (direita)



Fonte: do autor

O resultado desse passo é a divisão do resultado obtido da permutação IP em duas metades, L (*left*) e R (*right*) (figura 38), que serão enfim passados por parâmetro para a execução da função f_K .

3.3.4 Passo f_K - Função que usa uma chave

Figura 39 – Passo f_K - Função que usa uma chave

f_{K_1} - Função que usa a chave K_1

A função f_K é o componente mais complexo da execução do algoritmo e consiste de uma combinação de permutações e substituições e será chamada duas vezes durante o fluxo de execução, sendo uma vez para cada chave (K_1 e K_2). Como estamos criptografando, a primeira execução da função f_K deverá se utilizar da chave K_1 . A função f_K é definida por:

$$f_K(L, R) = (L \oplus F(R, SK), R)$$

A função f_K se utiliza da função F que por sua vez é definida por uma sequência de passos. Vamos executar iniciando da função mais interna até a mais externa e analisar cada parte.

Fonte: do autor

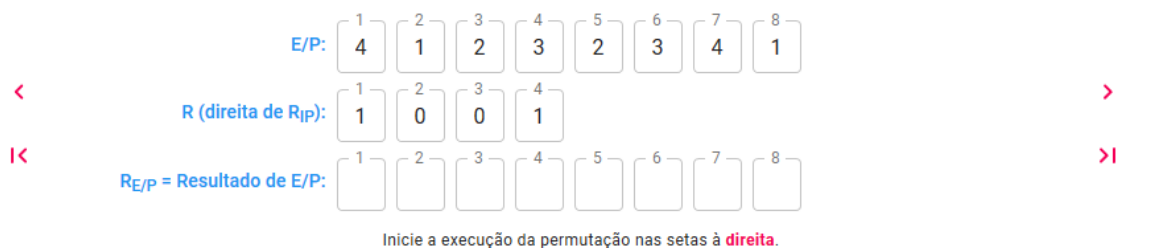
Esse passo é o único passo que se repete durante a execução da criptografia ou descriptografia. Essa função (figura 39) recebe 2 parâmetros de entrada. Um é uma sequência de 8 bits divididos entre L & R (4 bits na esquerda L e 4 bits na direita R) e o segundo parâmetro é a chave. Caso a execução escolhida seja a criptografia, a primeira vez que essa função é executada a chave utilizada será a K_1 e na segunda vez será a K_2 . Caso a execução escolhida seja a descriptografia, a ordem de utilização das chaves será inversa. A interface explica a tais peculiaridades e apresenta a função matemática desta.

Nesse passo estão concentradas todas as etapas da execução da função f_K . A etapas contidas nesse passo estavam inicialmente divididas em 2 passos. Um indo até o primeiro XOR e outro indo até o segundo XOR. Mas por esse passo se repetir optou-se por deixar todas as etapas em um único passo.

3.3.4.1 Etapa E/P - Permutação de Expansão

Figura 40 – Etapa E/P - Permutação de Expansão

E/P - Permutação de Expansão



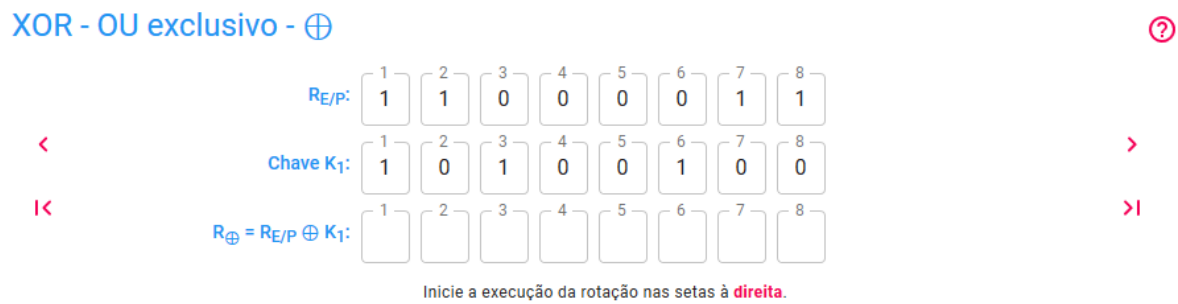
Fonte: do autor

A interface da etapa E/P (figura 40), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação,

apresenta a função de permutação E/P (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exhibe um componente capaz de executar passo a passo a função de permutação E/P. O parâmetro de entrada da permutação E/P é o parâmetro de entrada R da função f_K (4 bits do lado direito dos 8 bits recebidos de entrada).

3.3.4.2 Etapa XOR - OU exclusivo com K_1

Figura 41 – Etapa XOR - OU exclusivo com K_1



Fonte: do autor

A interface da primeira etapa XOR (figura 41), na região de explicação, descreve o que ocorre nessa etapa e exhibe um componente capaz de executar bit a bit a operação XOR (OU exclusivo) para obtenção do resultado dessa etapa. Os parâmetros de entrada dessa etapa são: 1. O resultado da permutação de expansão E/P e 2. Ou a chave K_1 ou a chave K_2 . Caso a execução escolhida seja a criptografia, a primeira vez que essa etapa é executada a chave utilizada será a K_1 e na segunda vez será a K_2 . Caso a execução escolhida seja a descriptografia, a ordem de utilização das chaves será inversa. A interface reflete essas particularidades em cada fluxo de execução.

3.3.4.3 Etapa S0 & S1 - Substituições S0 e S1

Figura 42 – Etapa S0 & S1 - Substituições S0 e S1

S0 & S1 - Substituições S0 e S1

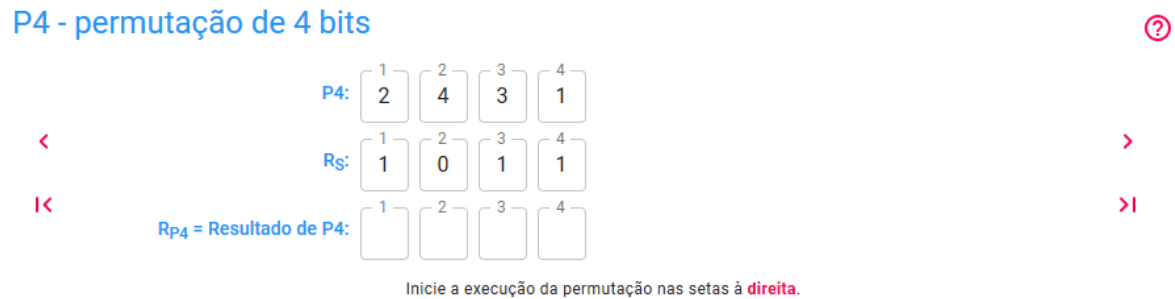


Fonte: do autor

A interface da etapa das substituições S0 e S1 (figura 42), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de um substituição, apresenta a obtenção dos parâmetros de entrada para cada substituição e explica como uma substituição deve ser interpretada. Após tal contextualização se exibe, para cada substituição, um componente capaz de executar passo a passo a substituição. Os parâmetros de entrada das substituições S0 e S1 são, respectivamente, as metades esquerda e direita do resultado obtido na etapa anterior (XOR com K_1). O resultado desta etapa é então a junção das metades após suas substituições individuais.

3.3.4.4 Etapa P4 - Permutação de 4 bits

Figura 43 – Etapa P4 - Permutação de 4 bits

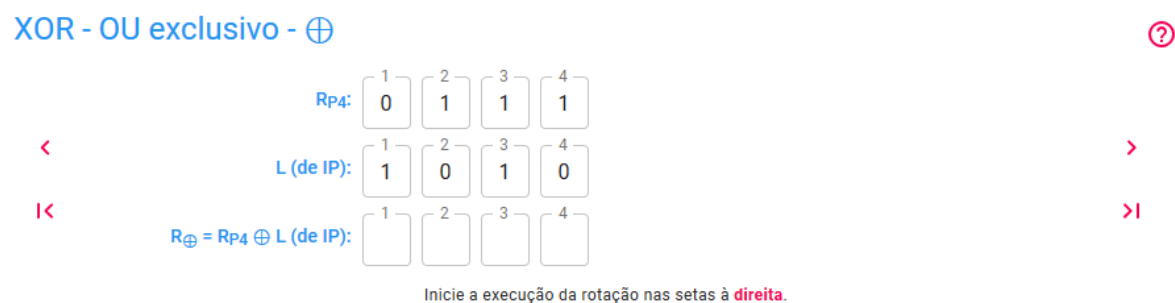


Fonte: do autor

A interface da etapa P4 (figura 43), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação P4 (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação P4. O parâmetro de entrada da permutação P4 é o resultado obtido na etapa anterior (Substituições S0 e S1). O resultado dessa execução será o parâmetro de entrada para a próxima etapa, XOR com L.

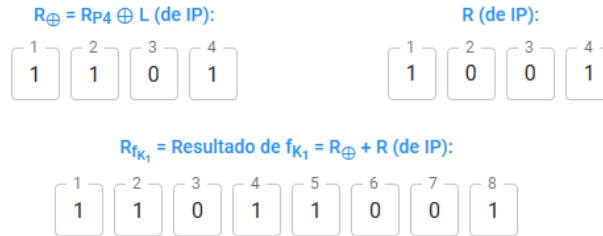
3.3.4.5 Etapa XOR - OU exclusivo com L

Figura 44 – Etapa XOR - OU exclusivo com L



Fonte: do autor

A interface da segunda etapa XOR (figura 44), na região de explicação, descreve o que ocorre nessa etapa e exibe um componente capaz de executar bit a bit a operação XOR (OU exclusivo) para obtenção do resultado dessa etapa. Os parâmetros de entrada dessa etapa são: 1. O resultado da permutação P4; e 2. L (metade esquerda do parâmetro de entrada de f_K). Na primeira vez que f_K é executada L será a esquerda do resultado da permutação inicial (IP) e na segunda vez L será a esquerda do resultado da Troca (SW).

3.3.4.6 Resultado de f_K Figura 45 – Resultado de f_K Resultado de f_{K_1} 

Fonte: do autor

A interface do resultado de f_K (figura 45) exibe as metades que compõem o resultado da função e a junção destes. A metade esquerda desse resultado sempre será o resultado da etapa anterior, XOR. A metade direita será a metade direita do parâmetro de entrada da função f_K . Que na primeira vez é a metade direita do resultado da permutação inicial (IP) e na segunda vez é a metade direita da Troca (SW).

3.3.5 Passo SW - Troca

Figura 46 – Passo SW - Troca

SW - Troca



Fonte: do autor

A interface do passo Troca (SW) (figura 46), na região de explicação, explica a definição da troca, apresenta a função de troca (incluindo função matemática) e explica

como esta deve ser interpretada. Este é o passo intermediário entre as execuções das funções f_K . Ela recebe por parâmetro o resultado da primeira execução da função f_K e o resultado deste passo é o parâmetro de entrada para a segunda execução da função f_K juntamente com a chave que não foi utilizada pela primeira execução.

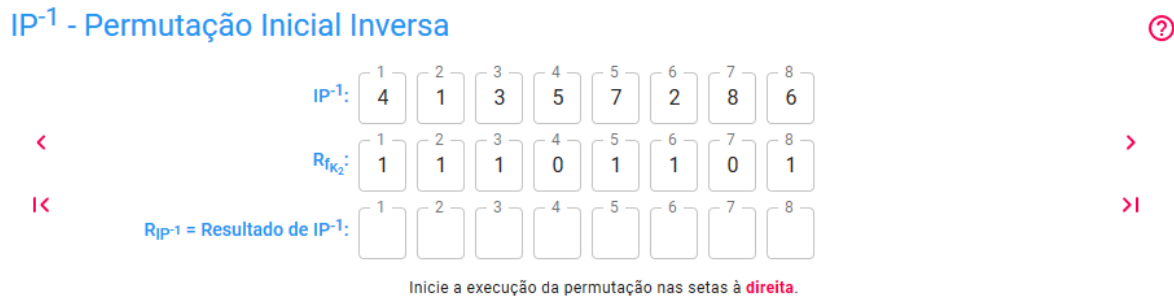
3.3.6 Passo f_K - Função que usa a outra chave

A segunda execução da função f_K é sequencialmente idêntica à primeira logo as interfaces de cada etapa são similares. As únicas diferenças são:

- O parâmetro de entrada é o resultado do passo Troca (SW) e não o resultado do passo Permutação inicial (IP) como ocorre na primeira execução da função.
- A chave utilizada é a chave que não foi utilizada na primeira execução da função f_K . Na criptografia a primeira execução da função f_K utiliza a chave K_1 e a segunda execução a chave K_2 . Já na descryptografia o inverso é verdade. A interface identifica o fluxo que está sendo executado e reflete essas diferenças tanto nas *labels* quanto nas explicações das etapas.

3.3.7 Passo IP^{-1} - Permutação Inicial Inversa

Figura 47 – Passo IP^{-1} - Permutação Inicial Inversa



Fonte: do autor

A interface da etapa IP^{-1} (figura 47), na região de explicação, contextualiza o objetivo pelo qual essa etapa existe, explica a definição de uma função de permutação, apresenta a função de permutação IP^{-1} (incluindo função matemática) e explica como esta deve ser interpretada. Após tal contextualização se exibe um componente capaz de executar passo a passo a função de permutação IP^{-1} . O parâmetro de entrada da permutação IP^{-1} é o resultado obtido no passo anterior (f_K).

3.3.8 Tela final - Resultado

Figura 48 – Resultado

Resultado da criptografia S-DES



Bits da mensagem:	1	2	3	4	5	6	7	8	=	Char
	0	1	1	1	0	0	1	0		r
Bits da chave:	1	2	3	4	5	6	7	8	9	10
	1	0	1	0	0	0	0	0	1	0
Mensagem cifrada:	1	2	3	4	5	6	7	8	=	Char
	0	1	1	1	0	1	1	1		w

Fonte: do autor

A interface do último passo (figura 48) exibe os dois parâmetros de entrada da execução, a Mensagem e a Chave, e o resultado da execução da criptografia ou descryptografia.

Figura 49 – *Feedback request*

CryptoEdu



O Código fonte desse simulador está disponível no GitHub:



Peço, encarecidamente, que responda à um questionário relativo à utilização do simulador:



São somente 6 perguntas de múltipla escolha e demora, em média, 1 minuto para ser respondido.

Este simulador foi desenvolvido com o objetivo de auxiliar o ensino da criptografia. E responder ao formulário auxilia a evolução deste.

Fonte: do autor

O último ponto na interface (figura 49) é uma solicitação do preenchimento do questionário de utilização do simulador. É exibido também um link para o repositório *GitHub* onde se encontra o simulador. No *GitHub* também é possível baixar a última versão publicada deste TCC.

3.4 Comparação com outros simuladores criptográficos

Os simuladores foram comparados em alguns pontos: **Idioma**; **Algoritmos**, quais algoritmos o simulador é capaz de simular; **Educativo**, se o simulador é voltado ao ensino

ou não; **I/O**, nível de detalhamento das entradas e saídas, em que, **algoritmo** exibe somente as entradas e saídas do algoritmo, **passo** exibe também as entradas e saídas de cada passo do algoritmo e **etapa** exibe também as entradas e saídas de cada etapa dentro de cada passo; **Open Source**, se o simulador possui código aberto e permite ou não que colaboradores estendam suas funcionalidades; **Plataforma**; e **Responsivo**, que indica o nível de responsividade da interface do simulador.

Comparando o simulador desenvolvido com os outros simuladores criptográficos encontrados, levando em consideração os pontos já explícitos, temos:

	CyberChef	CrypTool	S-DES Sim. App	S-DES Sim. <i>online</i>	S-DES Sim. Win	CryptoEdu
Idioma	Inglês	Inglês	Inglês	Koreano	Inglês	Português
Algoritmos	Vários	Vários	S-DES	S-DES	S-DES	S-DES
Educativo	Não	Sim	Sim	Sim	Sim	Sim
I/O	algoritmo	algoritmo	passo	passo	passo	etapa
Open Source	Sim	Não	Não	Sim	Não	Sim
Plataforma	<i>online</i>	SO's	Android	<i>online</i>	Windows	<i>online</i>
Responsivo	<i>desktop</i>	<i>desktop</i>	<i>mobile</i>	<i>desktop</i>	<i>desktop</i>	<i>desktop e mobile</i>

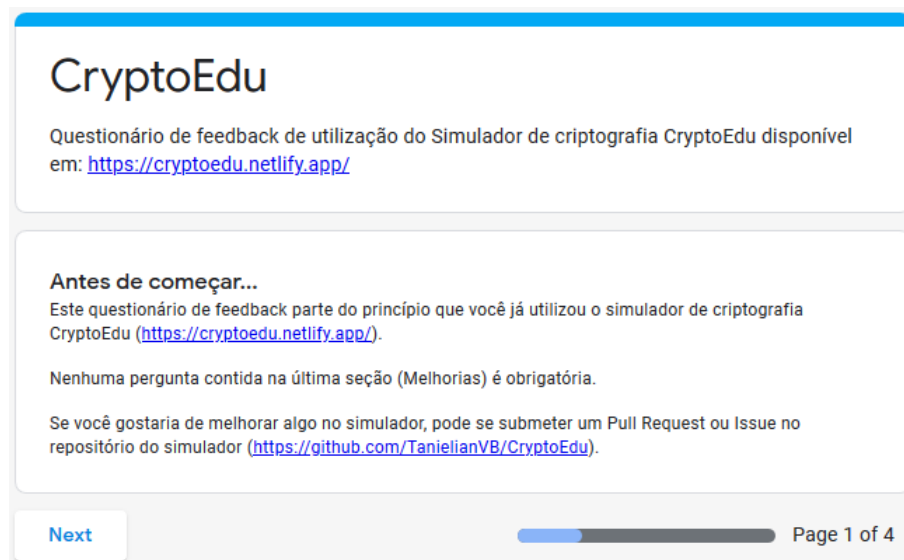
Foram marcados de verde, na tabela, os pontos dos simuladores que se sobressaem positivamente quando analisados em um contexto de acessibilidade e ensino-aprendizagem.

4 Pesquisa de reação

Com o intuito de mensurar de forma objetiva o alcance dos objetivos propostos neste trabalho, foi proposta uma pesquisa de reação sobre a utilização do simulador. Essa pesquisa possui 6 perguntas, 1 relativa ao usuário, 3 relativas à absorção de conhecimento oriundas da utilização do simulador e 2 relativas à melhorias que podem ser feitas no simulador.

O questionário de *feedback* de utilização do simulador foi disponibilizado através da utilização do *Google Forms* através do link <https://forms.gle/f1DAgWvTyM2uJyLt9>.

Figura 50 – Início do questionário



A imagem mostra a interface inicial de um formulário no Google Forms. No topo, há um cabeçalho azul com o título "CryptoEdu". Abaixo dele, um texto informa que o "Questionário de feedback de utilização do Simulador de criptografia CryptoEdu" está disponível em <https://cryptoedu.netlify.app/>. A seção principal, intitulada "Antes de começar...", contém três parágrafos: o primeiro explica que o formulário parte do princípio de que o usuário já utilizou o simulador; o segundo afirma que nenhuma pergunta na última seção é obrigatória; o terceiro sugere que, caso o usuário queira melhorar algo, pode submeter um Pull Request ou Issue no repositório do simulador (<https://github.com/TanielianVB/CryptoEdu>). Na base da tela, há um botão "Next" à esquerda, uma barra de progresso no centro e o texto "Page 1 of 4" à direita.

Fonte: do autor

Antes de apresentar as questões para o usuário é apresentado essa tela de apresentação do questionário (figura 50).

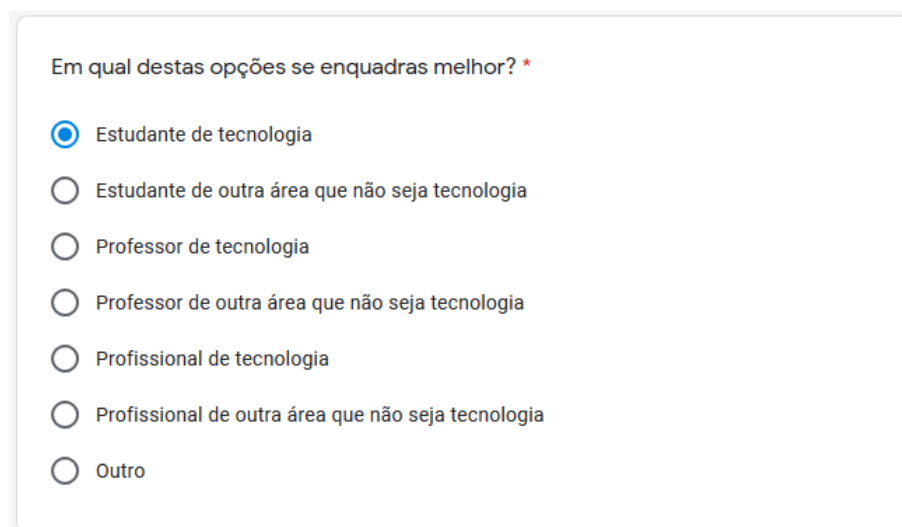
4.1 Perguntas

O questionário é composto de 6 perguntas, sendo 4 obrigatórias e 2 opcionais, que serão descritas à seguir.

Um fator que foi levado em consideração ao desenvolver o questionário é a dificuldade de adquirir respostas dos usuários. Tanto por esse motivo quando para facilitar a análise dos resultados se optou por se utilizar somente de perguntas de múltipla escolha.

4.1.1 Enquadramento do usuário

Figura 51 – Enquadramento do usuário



Em qual destas opções se enquadras melhor? *

- ☒ Estudante de tecnologia
- ☐ Estudante de outra área que não seja tecnologia
- ☐ Professor de tecnologia
- ☐ Professor de outra área que não seja tecnologia
- ☐ Profissional de tecnologia
- ☐ Profissional de outra área que não seja tecnologia
- ☐ Outro

Fonte: do autor

Esta pergunta (figura 51) tem como objetivo ter uma *baseline* do tipo de usuário que está respondendo o questionário e dessa forma ser capaz de obter uma curva de crescimento por 'perfil'.

4.1.2 Conhecimento antes e depois da utilização do simulador

As próximas 2 perguntas objetivam mensurar se houve evolução do nível de conhecimento dos usuários sobre os processos de criptografia presentes na simulação do algoritmo S-DES. O escopo das perguntas envolve os processos (permutação, rotação, substituição, xor e troca) presentes no algoritmo ao invés das etapas (P10, LS-1, P8, LS-2, IP, etc...) pois munido do conhecimento dos processos todas as etapas podem ser reproduzidas.

Figura 52 – Conhecimento antes da utilização do simulador

Como você quantificaria o seu nível de conhecimento, de forma geral, nos processos envolvidos no algoritmo S-DES (permutação, rotação, substituição, etc.) ANTES da utilização do simulador? *

- ☐ 0 - Não tinha nenhum conhecimento.
- ☒ 1 - Tinha pouco conhecimento de algum ou alguns dos processos.
- ☐ 2 - Tinha médio conhecimento de algum ou alguns dos processos.
- ☐ 3 - Tinha muito conhecimento de algum ou alguns dos processos.
- ☐ 4 - Tinha conhecimento completo de todos os processos.

Fonte: do autor

Para mensurar se houve evolução do nível de conhecimento é questionado primeiramente como o usuário quantificaria o nível de conhecimento, de maneira geral, que ele possui nos processos presentes no algoritmo antes da utilização do simulador (figura 52). Buscando obter o ponto de partida do nível de conhecimento.

Figura 53 – Conhecimento depois da utilização do simulador

Como você quantificaria o seu nível de conhecimento, de forma geral, nos processos envolvidos no algoritmo S-DES (permutação, rotação, substituição, etc.) DEPOIS da utilização do simulador? *

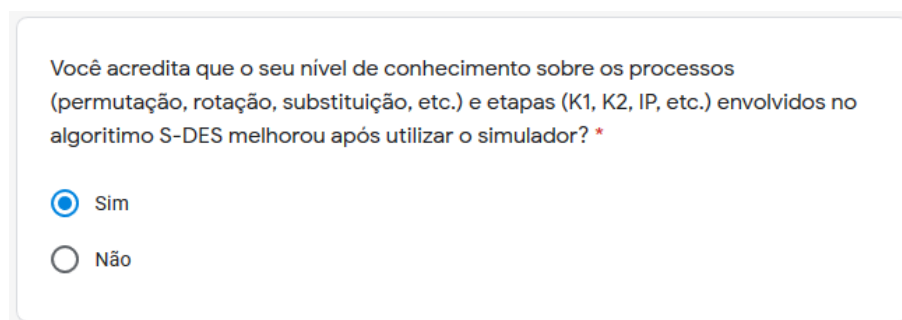
- ☐ 0 - Não tem nenhum conhecimento.
- ☐ 1 - Tem pouco conhecimento de algum ou alguns dos processos.
- ☐ 2 - Tem médio conhecimento de algum ou alguns dos processos.
- ☒ 3 - Tem muito conhecimento de algum ou alguns dos processos.
- ☐ 4 - Tem conhecimento completo de todos os processos.

Fonte: do autor

A próxima pergunta indaga esse mesmo nível de conhecimento após a utilização do simulador (figura 53). Buscando obter o ponto de chegada do nível de conhecimento.

4.1.3 Efetividade do aprendizado

Figura 54 – Efetividade do aprendizado



Você acredita que o seu nível de conhecimento sobre os processos (permutação, rotação, substituição, etc.) e etapas (K1, K2, IP, etc.) envolvidos no algoritmo S-DES melhorou após utilizar o simulador? *

☒ Sim

☐ Não

Fonte: do autor

Esta pergunta (figura 54) tem como objetivo mensurar, de maneira absoluta, se existiu ganho de conhecimento por parte do usuário.

4.1.4 Melhorias no simulador

Com as próximas 2 perguntas tem-se como objetivo ser capaz de identificar quais etapas explícitas no simulador precisam ser melhoradas. O escopo das perguntas envolve as etapas (P10, LS-1, P8, LS-2, IP, etc...) presentes no algoritmo ao invés dos processos (permutação, rotação, substituição, xor e troca) pois cada descrição e execução é singular àquela determinada etapa.

Figura 55 – Melhorias nas descrições das etapas

Quais das etapas poderiam ter sua descrição melhoradas? Marque todas aplicáveis.

- ☐ P10 - Permutação de 10 bits
- ☐ LS-1 - Rotação para a esquerda de 1 posição
- ☐ P8 - Permutação de 8 bits
- ☐ K1 - Obtenção da primeira chave
- ☐ LS-2 - Rotação para a esquerda de 2 posições
- ☐ K2 - Obtenção da segunda chave
- ☐ IP - Permutação Inicial
- ☐ E/P - Permutação de expansão
- ☐ XOR - OU exclusivo
- ☐ S0 & S1 - Substituições
- ☐ P4 - Permutação de 4 bits
- ☐ SW - Swap
- ☐ IP-1 - Permutação Inicial inversa

Fonte: do autor

A primeira pergunta (figura 55) dessa seção destina-se à identificar quais etapas da execução do algoritmo possuem explicações que podem ser melhoradas, ou seja, não foram suficiente para compreensão completa da etapa pelo usuário.

Figura 56 – Melhorias nas execuções das etapas

Quais das etapas poderiam ter sua execução melhoradas? Marque todas aplicáveis.

- ☐ P10 - Permutação de 10 bits
- ☐ LS-1 - Circular Left Shift de 1 posição
- ☐ P8 - Permutação de 8 bits
- ☐ K1 - Obtenção da primeira chave
- ☐ LS-2 - Circular Left Shift de 2 posições
- ☐ K2 - Obtenção da segunda chave
- ☐ IP - Permutação Inicial
- ☐ E/P - Permutação de expansão
- ☐ XOR - OU exclusivo
- ☐ S0 & S1 - Substituições
- ☐ P4 - Permutação de 4 bits
- ☐ SW - Swap
- ☐ IP-1 - Permutação Inicial inversa

Fonte: do autor

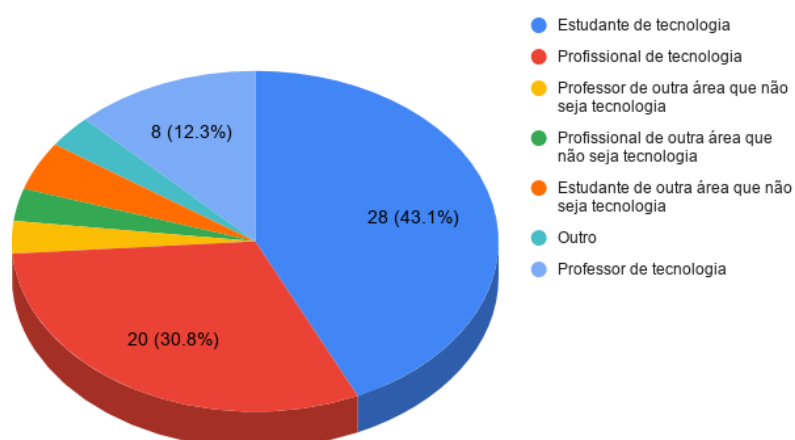
A segunda pergunta (figura 56) dessa seção destina-se à identificar quais etapas da execução do algoritmo possuem execuções (passo a passo) que podem ser melhoradas, ou seja, não foram suficiente para compreensão completa do passo a passo da etapa pelo usuário.

4.2 Resultados da pesquisa

Essa seção descreve os resultados da pesquisa realizada. A pesquisa contou com a participação de alunos e professores da FBuni e da UNIFOR, como também de profissionais das seguintes empresas: Unimake, RCN e Fortes Tecnologia.

4.2.1 Enquadramento do usuário

Figura 57 – Enquadramento do usuário

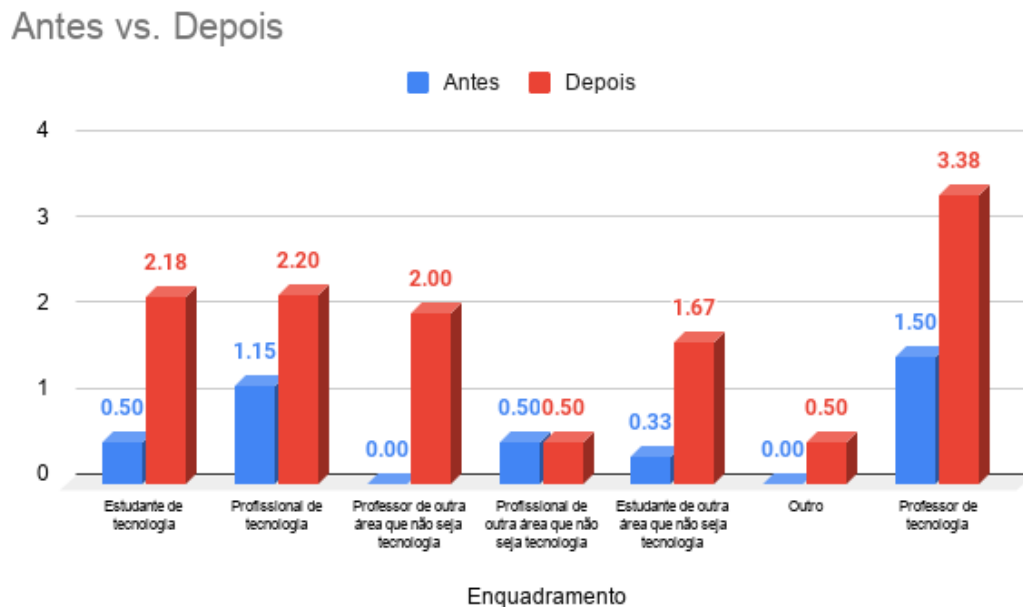


Fonte: do autor

Foram obtidas 65 respostas. Sendo 86,2% da área tecnológica onde 43,1% são estudantes, 30,8% profissionais e 12,3% professores (figura 57). Embora esse montante não seja o suficiente para um estudo do ponto de vista estatístico, é suficiente para mensurar a efetividade da ferramenta desenvolvida.

4.2.2 Conhecimento antes e depois da utilização do simulador

Figura 58 – Conhecimento antes vs. depois da utilização do simulador por enquadramento



Fonte: do autor

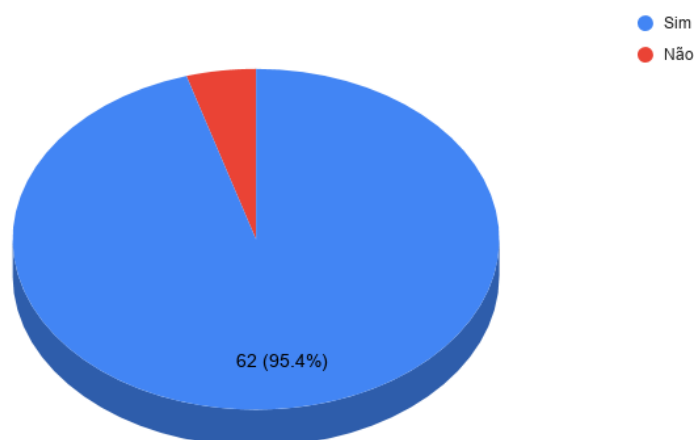
Analisando o nível de conhecimento antes e depois para cada enquadramento (figura 58) temos:

- Estudantes de tecnologia apresentaram um aumento médio de 1,68 pontos. A média de conhecimento antes era entre **nenhum** e **pouco** e após a utilização do simulador passou a ser **médio** ou superior.
- Profissionais de tecnologia apresentaram um aumento médio de 1,05 pontos. A média de conhecimento antes era **pouco** e após a utilização do simulador passou a ser **médio** ou superior.
- Professores de tecnologia apresentaram um aumento médio de 1,88 pontos. A média de conhecimento antes era entre **pouco** e **médio** e após a utilização do simulador passou a ser **muito** ou superior.

Infelizmente, não foram obtidos dados suficientes dos outros enquadramentos que tornasse viável extrair qualquer tipo de conclusão sobre estes.

4.2.3 Efetividade do aprendizado

Figura 59 – Efetividade do aprendizado

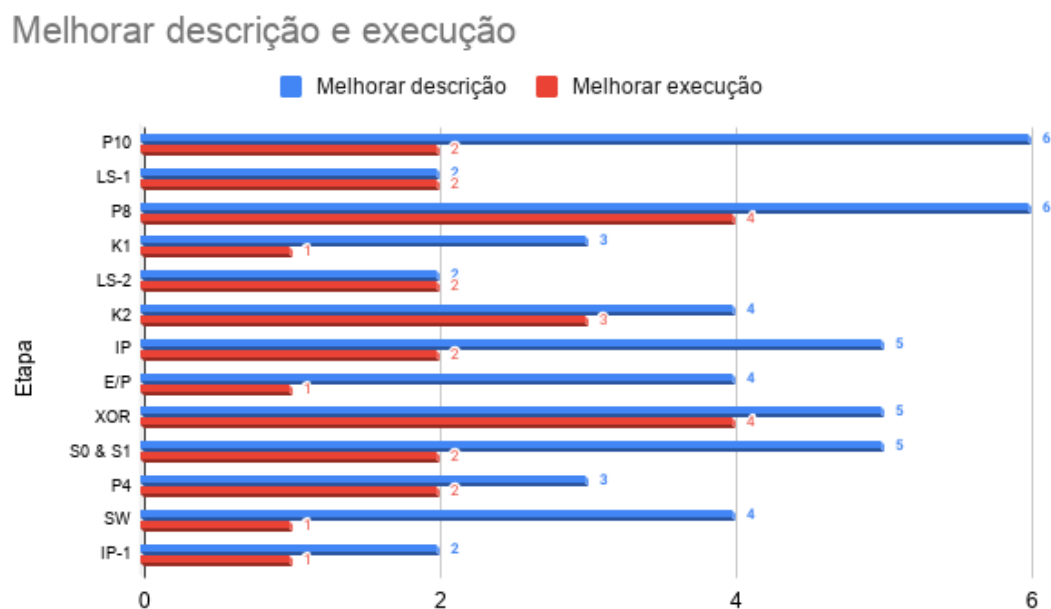


Fonte: do autor

95,4% dos entrevistados consideram que o seu nível de conhecimento sobre os processos e etapas apresentados no simulador melhorou após a utilização do simulador (figura 59). Um ponto interessante é que mesmo alguns entrevistados que se mensuraram no mesmo nível de conhecimento antes e depois da utilização do simulador ainda assim alegaram ter aprendido com o simulador.

4.2.4 Melhorias no simulador

Figura 60 – Melhorias no simulador



Fonte: do autor

Somente 21,53% dos entrevistados apontaram que algumas etapas poderiam ser melhoradas (figura 60). Dentre os pontos de melhoria apontados as descrições das permutações e substituição foram os mais indicados como pontos que precisam de melhoria.

5 Conclusões e Trabalhos futuros

Esta pesquisa abordou a temática de Objeto de Aprendizagem (OA) da Informática na Educação.

Inicialmente, levantou-se os desafios do processo de ensino-aprendizagem da criptografia no meio acadêmico, ressaltando a necessidade desse conhecimento para o formando, independentemente de seguir carreira acadêmica ou de TI. A complexidade dos algoritmos aliada ao tempo normalmente disponível ao ensino desses são elementos que dificultam o processo de ensino-aprendizagem, mas isso pode ser mitigado com ferramentas voltadas ao ensino criptográfico.

Vale ressaltar o aumento contínuo do uso dos OAs no meio acadêmico em razão das vantagens provenientes de seu uso no processo ensino-aprendizagem, como: o auxílio de ensino presencial e principalmente à distância (EaD), sua acessibilidade, e grande interação entre o usuário e o OA. Como exemplo de OA, temos o jogo *Minecraft: Education Edition* que já foi aplicado no ensino de Biologia, Ecologia, Física, Química, Geologia, Geografia e até Cibersegurança.

O simulador resultado da pesquisa realizada trás avanços e contribuições para a área da Informática Educacional, dos quais destacam-se: é em português; é o primeiro a apresentar explicação e execução passo a passo de todas as etapas, de cada passo, envolvidas tanto no processo de criptografia como no processo de descryptografia; e é mais acessível pois pode ser visualizado tanto em *desktop* como em *mobile*.

O simulador foi disponibilizado para uma comunidade de alunos e professores a fim de avaliar, por meio de uma pesquisa, a sua efetividade no processo de ensino-aprendizagem da técnica de criptografia por cifra de blocos. Esta efetividade se fez comprovada quando, após a utilização do simulador, **95,4%** dos entrevistados consideraram que melhoraram o seu nível de conhecimento sobre os processos e etapas nele apresentados. Além disso, alunos de tecnologia passaram a ter nível de conhecimento **médio** sobre esses mesmos processos e etapas quando o seu nível de conhecimento original era **nenhum** ou **pouco**.

Como possíveis trabalhos futuros, sugere-se: aplicar ao simulador melhorias nos aspectos escolhidos pelos entrevistados como pontos que podem ser melhorados, tornando, assim, o simulador ainda mais eficiente no ensino-aprendizagem; e adicionar outros algoritmos ao simulador, dessa forma, aumentando seu nicho de aplicabilidade.

Referências

- ABUZAID, A. *et al.* The design and implementation of a cryptographic education tool. In: . [S.l.: s.n.], 2011. v. 1, p. 193–198. 19
- AUDINO, D. F. Objetos de aprendizagem hipermídia aplicado à cartografia escolar no sexto ano do ensino fundamental em geografia. 57 p. Dissertação (Mestrado) — Universidade Federal de Santa Catarina - UFSC, 2012. 17
- AUSUBEL, D. P.; NOVAK, J. D.; HANESIAN, H. Psicologia educacional. 2. ed. [S.l.]: Interamericana, 1980. ISBN 8520100848. 15, 16
- AVELINO, D.; AVELINO, I. C. Aplicações da criptografia em ambientes computacionais. In: . [S.l.]: IV SEGeT – Simpósio de Excelência em Gestão e Tecnologia, 2007. 25
- BANKS, J.; II, J. S. C.; NELSON, B. L. Discrete-Event System Simulation. 5. ed. [S.l.]: Prentice Hall, 2009. ISBN 9780136062127. 18
- BARBOSA, G.; SCORTEGAGNA, L. O uso de objetos de aprendizagem na educação financeira: Metodologia para avaliação pautada nos aspectos tecnológicos e pedagógicos. Boletim GEPEM, 2015. ISSN 2176-2988. 13
- BELMONTE, V.; GROSSI, M. G. R. Ambientes virtuais de aprendizagem: Um panorama da produção nacional. CEFET-MG - Centro Federal de Educação Tecnológica de Minas Gerais, 2010. 17
- BRAGA, J. C. Objetos de Aprendizagem Volume 1 - Introdução e Fundamentos. [S.l.]: Editora UFABC, 2014. ISBN 9788568576038. 17
- BRAGA, J. C. Objetos de Aprendizagem Volume 2 - Metodologia de Desenvolvimento. [S.l.]: Editora UFABC, 2015. ISBN 9788568576045. 17
- BROCARD, M. L.; ROLT, C. R. D.; FERNANDES, R. Introdução à certificação digital: da criptografia ao carimbo de tempo. BRy Tecnologia, 2006. 26
- CASTOLDI, R.; POLINARSKI, C. A. A utilização de recursos didático-pedagógicos na motivação da aprendizagem. I Simpósio Nacional de Ensino de Ciência e Tecnologia, 2009. 16
- FERREIRA, S. M. M. Os recursos didáticos no processo de ensino-aprendizagem - estudo de caso da escola secundária cónego jacinto. Universidade Jean Piaget de Cabo Verde, 2007. 16
- GAINES, H. F. Cryptanalysis: a study of ciphers and their solution. [S.l.]: Dover Publications, 1956. ISBN 9780486200972. 25
- GARMPIS, A. Design and development of a web-based interactive software tool for teaching operating systems. Journal of Information Technology Education, v. 10, 2011. 13
- GCHQ *Government Communications Headquarters.* CyberChef. 2020. Acessado em 28/11/2020. Disponível em: <<https://github.com/gchq/CyberChef/>>. 19

- GEARY, J.; RONKE, T.; GEARY, M. Using minecraft education edition to teach cybersecurity self-defense. In: . [S.l.: s.n.], 2019. v. 9. 17
- HAMAWAKI, M. H.; PELEGRINI, C. de M. As ferramentas do ensino a distância e suas contribuições para a eficácia no processo de aprendizagem do aluno. CEPPG - n° 21, p. 84 à 91, 2009. ISSN 1517-847. 15, 18
- ITU, I. T. U. Security architecture for open systems interconnection for CCITT applications: Recommendation X.800. [S.l.]: The international telegraph and telephone consultative committee - CCITT, 1991. Geneva. 25
- KENSKI, V. M. Educação e Tecnologias: o novo ritmo da educação. [S.l.]: Papirus Editora, 2007. ISBN 9788530808280. 16
- KIOKI, E. Y.; SANTIAGO, P. P.; SOARES, A. C. Um simulador didático como ferramenta de apoio ao ensino da disciplina de sistemas operacionais. Revista INICIA, v. 8, 2008. 13
- KNUDSEN, J. Java Cryptography. [S.l.]: O'Reilly, 1998. 25
- LOPES Átila R. MEMO: Software de apoio didático para o ensino de gerência de memória. Dissertação (Mestrado) — Universidade Estadual do Ceará - UECE, 2012. 13
- LUBURIĆ, N. *et al.* Crypto-tutor: An educational tool for learning modern cryptography. In: . [S.l.: s.n.], 2016. p. 205–210. ISBN 9781509028665. 19
- MAIA, L. P. SOsim: Simulador para o ensino de sistemas operacionais. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro - UFRJ, 2001. 13, 18
- MAIA, L. P.; PACHECO, A. C. A simulator supporting lectures on operating systems. 33'd ASEE/IEEE Frontiers in Education Conference, 2003. 13, 18
- MOUNTOGIANNAKIS, A. G. S-DES Simulator. 2015. Acessado em 28/11/2020. Disponível em: <https://play.google.com/store/apps/details?id=com.sdesandroid&hl=en_CA>. 22
- PARRA, N. Didática: dos modelos a prática do ensino. Cadernos de Didática, p. 1–18, 1985. 16
- PERMADI, E. S-DES Simulator. 2018. Acessado em 28/11/2020. Disponível em: <<https://des-simulator.software.informer.com/>>. 24
- PIROPO, B. Enigma (máquina). 2007. Acessado em 16/12/2020. Disponível em: <<https://www.bpiropo.com.br/fpc20071203.htm>>. 28, 29, 30
- SCHAEFER, E. F. A simplified data encryption standard algorithm. Cryptologia, Taylor & Francis, v. 20, n. 1, p. 77–84, Junho 1996. Disponível em: <<https://doi.org/10.1080/0161-119691884799>>. 34
- SHORT, D. Teaching scientific concepts using a virtual world - minecraft. In: . [S.l.: s.n.], 2012. v. 58. 17
- SILVA, I. de O.; JUNIOR, A. R. P. Criptografia quântica: Uma alternativa segura? Faculdade de Tecnologia de Ourinhos – FATEC, 2012. 13

- SILVA, M. J. da *et al.* A utilização do modelo “adedonha” como recurso didático-pedagógico para o ensino-aprendizagem. V Congresso Nacional de Educação, Realize Editora, 2018. ISSN 1517-847. 16
- SILVA, R. Aes - advanced encryption standard. Instituto de Engenharia de Sistemas e Computadores – Investigação e Desenvolvimento (Lisboa), 2009. 13
- SOUZA, S. E. de. O uso de recursos didáticos no ensino escolar. I Encontro de Pesquisa em Educação, Arq Mudi, 2007. 16
- STALLINGS, W. Cryptography and Network Security: Principles and Practices. [S.l.]: Prentice Hall, 2010. ISBN 0136097049. 31, 34, 35, 36, 37
- STALLINGS, W. Criptografia e Segurança de Redes: Princípios e Práticas. [S.l.]: Pearson, 2014. 25, 28, 29, 31, 32, 34
- TANENBAUM, A. S. Redes de computadores. [S.l.]: Editora Campus, 2003. 25, 28, 29, 31, 32
- CrypTool Contributors. CrypTool 2. 2014. Acessado em 28/11/2020. Disponível em: <<https://www.cryptool.org/en/ct2/>>. 20
- CrypTool Contributors. JCrypTool. 2016. Acessado em 28/11/2020. Disponível em: <<https://www.cryptool.org/en/jct/>>. 21
- CrypTool Contributors. CrypTool. 2020. Acessado em 28/11/2020. Disponível em: <<https://www.cryptool.org/>>. 20
- TRIVELATO, S. L. F.; OLIVEIRA, O. B. de. Prática docente: o que pensam os professores de ciências biológicas em formação. Políticas educacionais, tecnologias e formação do educador: repercussões sobre a didática e as práticas de ensino, XIII Encontro Nacional de Didática e Prática de Ensino - ENDIPE, 2006. 16
- WIKIPÉDIA. Enigma (máquina). Wikipédia, 2020. Acessado em 28/11/2020. Disponível em: <[https://pt.wikipedia.org/wiki/Enigma_\(máquina\)](https://pt.wikipedia.org/wiki/Enigma_(máquina))>. 27
- WILEY, D. The ASTD e-learning handbook: Learning objects need instructional design theory. McGraw-Hill, p. 115–126, 2002. 17
- WOO, J. G. S-DES Simulator. 2018. Acessado em 28/11/2020. Disponível em: <<https://devgw.github.io/S-DES-simulator/index.html>>. 23
- YOUNG, A.; YUNG, M. Malicious Cryptography: Exposing Cryptovirology. [S.l.]: Wiley Publishing, Inc., 2004. ISBN 9780764549755. 25